Aalto University
School of Engineering

Santeri Myllynen

Utilization of Artificial Intelligence in the Analysis of Nuclear Power Plant Requirements

Master's thesis submitted in a partial fulfillment of the requirements for the degree of Master of Science in Technology.

Espoo 8.4.2019
Supervisor: Sanna Syri, Professor
Advisors: Satu Sipola, M.Sc. (Tech)
Pekka Nuutinen, M.Sc. (Tech)

Aalto University
School of Engineering

Aalto University, P.O. BOX 11000, 00076
AALTO
www.aalto.fi

Abstract of master's thesis

Author Santeri Myllynen

Title of thesis Utilization of Artificial Intelligence in the Analysis of Nuclear Power Plant Requirements

Master programme Master's Programme in Advanced Energy Solutions **Code** ENG30

Major Sustainable Energy Conversion Processes

Thesis supervisor Sanna Syri, Professor

Thesis advisors Satu Sipola, M.Sc. (Tech), Pekka Nuutinen, M.Sc. (Tech)

Date 08.04.2019 **Number of pages** 101 + 1 **Language** English

Abstract

Nuclear power plant projects are often characterized by two factors: they are time-consuming and capital-intensive. These current challenges include descriptive and non-harmonized requirements demanded in the nuclear power industry resulting in the adaptation to a new licensing domain being very data-intensive, laborious, and tardy. Furthermore, the sheer volume of these requirements also poses a challenge. Nevertheless, by utilizing artificial intelligence in the analysis of nuclear power plant requirements, licensing and engineering could be facilitated and errors reduced in the allocation of requirements.

This Master's thesis develops an algorithm capable of recognizing natural language to classify nuclear power plant requirements into predefined categories by utilizing supervised machine learning. The study was performed in close cooperation with an AI company, Selko Technologies Oy, being responsible for the development of the algorithm based on the classified set of requirements and the needs of Fortum.

The algorithm consists of a nuclear power industry-specific language model involving a long short-term memory network, and a classifier based on a feedforward neural network. The language model and classifier were trained by using the YVL Guides issued by the Finnish Radiation and Nuclear Safety Authority (STUK). For training the classifier, a small selection of the requirements were classified according to the two-level predefined hierarchy. The algorithm was tested on the selected YVL Guides and a set of requirements issued by the Office for Nuclear Regulation in United Kingdom.

The results include a predetermined requirements hierarchy, the content of the categories, natural language processing algorithm, requirements classified by both the experts and algorithm, and model accuracies in each test case. The accuracies of the classification tasks are promising indicating that the current methods are suitable for categorizing natural language as long as there is a qualified and sufficient amount of training data in place. The conclusions also suggest proceeding to research the capability of the models in other requirements analysis related tasks, such as atomizing long requirements and combining similar requirements into one.

Keywords Artificial Intelligence, Machine Learning, Systems Engineering, Requirements Analysis, Nuclear Power

Aalto-yliopisto
Insinööritieteiden korkeakoulu

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi

Diplomityön tiivistelmä

Tekijä Santeri Myllynen		
Työn nimi Tekoälyn hyödyntäminen ydinvoimalaitosvaatimusten analysoinnissa		
Koulutusohjelma Master's Programme in Advanced Energy Solutions		**Koodi** ENG30
Pääaine Sustainable Energy Conversion Processes		
Työn valvoja Professori Sanna Syri		
Työn ohjaajat DI Satu Sipola, DI Pekka Nuutinen		
Päivämäärä 08.04.2019	**Sivumäärä** 101 + 1	**Kieli** englanti

Tiivistelmä

Ydinvoimalaitosprojektit ovat usein pitkäkestoisia ja pääomaintensiivisiä. Yhtenä projektien ominaisena haasteena voidaan pitää suurta määrää kuvailevia ja epäyhtenäisiä vaatimuksia. Lisäksi ydinvoimalaitosdesignin vieminen ja suunnittelun sopeuttaminen uuteen lisensiointiympäristöön vaatii paljon tiedonhallintaa. Lisäksi se on työlästä ja hidasta. Tekoälyn hyödyntäminen ydinvoimalaitosvaatimusten analysoimisessa voisi nopeuttaa lisensiointi- ja suunnitteluprosesseja, sekä vähentää virheitä vaatimusten allokoinnissa.

Tässä diplomityössä on kehitetty luonnollisen kielen prosessointiin kykenevä algoritmi ydinvoimalaitosvaatimusten luokitteluun. Työssä vaatimukset on luokiteltu ennalta määrättyihin kategorioihin ohjattua koneoppimista hyödyntämällä. Tutkimus on tehty yhteistyössä tekoäly-yrityksen Selko Technologies Oy:n kanssa, joka on vastannut algoritmin kehittämisestä Fortumin toimittaman luokitellun vaatimusjoukon ja tarpeiden perusteella.

Algoritmi koostuu ydinvoima-alan kielimallista ja luokittelijasta. Kielimalli pohjautuu pitkään lyhytaikaisen muistin verkkoon ja luokittelija myötäkytkettyyn neuroverkkoon. Kielimallin ja luokittelijan kouluttamiseen on käytetty Suomen säteily- ja ydinturvallisuusviranomaisen Säteilyturvakeskuksen (STUK) Ydinturvallisuusohjeita. Luokittelijan kouluttamista varten tietty osa vaatimuksista on kategorisoitu kaksitasoisen ennalta määritellyn hierarkian mukaisesti. Algoritmin testaukseen on käytetty sekä valittua Ydinturvallisuusohjeiden vaatimusjoukkoa että Yhdistyneiden kuningaskuntien ydinturvallisuusviranomaisen (ONR) yhtä vaatimusjoukkoa.

Työn tuloksena syntyi ennalta määritetty vaatimushierarkia sekä luonnollista kieltä prosessoiva algoritmi. Lisäksi työssä määriteltiin, mitä asioita kuuluu eri vaatimusluokkiin. Määrittelyn jälkeen sekä asiantuntijat että algoritmi luokittelivat työssä käytetyn datan. Mallin tarkkuus ja käytettävyys pystyttiin testaamaan lopuksi testidatalla. Saadut tarkkuudet vaatimusten luokittelussa ovat lupaavia ja osoittavat, että nykyiset menetelmät soveltuvat hyvin luonnollisen kielen luokitteluun, mikäli vain koulutusdata on laadukasta ja sitä on riittävästi. Tutkimusta voitaisiin jatkaa kokeilemalla mallien soveltumista myös muissa vaatimusten analysointiin liittyvissä tehtävissä. Näitä ovat esimerkiksi pitkien vaatimusten pilkkominen lyhempiin ja selkeämmin määriteltyihin lauseisiin sekä samanlaisten vaatimusten yhdistäminen yhdeksi vaatimukseksi.

Avainsanat Tekoäly, Koneoppiminen, Systeemisuunnittelu, Vaatimusten Analysointi, Ydinvoima

Forewords

This Master's thesis was conducted for Fortum Power and Heat Oy in cooperation with the AI company, Selko Technologies Oy. This diverse research project has enabled me to widen my knowledge of two broad subjects: artificial intelligence and systems engineering. I am very pleased with this endeavor work which has required much effort.

I would like to humbly thank my kind advisors, Satu Sipola and Pekka Nuutinen, for this edifying opportunity, the trust shown me as well as the effort taken to instruct me at Fortum. I would also like to thank Professor Sanna Syri for supervising me. In addition, I am thankful for the helpful writing guidance provided by Ms. Anya Siddiqi via the Language Centre's Writing Clinic service.

This Master's thesis would not have been attainable without the close cooperation with Selko Technologies Oy, and especially CEO Tuomas Ritola and Data Scientist Aditya Jitta, who have shared their knowledge and supported me throughout the project. Thank you very much for the fruitful cooperation. Furthermore, this study has involved many Fortum experts with diversified competences. Huge thanks to all of you who have voluntarily taught me and provided valuable expertise. I wish to thank everyone for their kind assistance and time.

These six years of education at Aalto University have been interesting. I have learned more than I could have ever imagined. The great memories I possess of student life and the Otaniemi campus are due to the amazing and energetic people I have met. Thank you, one and all.

Work and life in general are a bit easier when there is a strong network supporting you. This support for which I am very grateful comes from my dear family: my mother and father as well as older sister and brother, including their families. I am very thankful for your kind support throughout my life.

In addition to my immediate family, I feel privileged to share my life with you, Sini. It has been a wonderful journey to walk alongside you, and I am enthusiastically looking forward to our shared future. Thank you for your love.

Finally, I would like to conclude my thoughts of the life up to this day and looking at the forthcoming marvel of life. As my respected and late grandfather stated as his last words:

We are here to learn.

Espoo, 8th of April 2019

Santeri Myllynen

Table of Contents

Abbreviations

ADLAS®	Advanced Licensing and Safety Engineering Method (Registered trademark of Fortum)
AI	Artificial Intelligence
ANN	Artificial Neural Network
BEATM	Both-Ends-against-the-Middle
CI	Configuration Item
CNN	Convolutional Neural Network
EARS	The Easy Approach to Requirements Syntax
FOAK	First-of-a-kind
HAEA	Hungarian Atomic Energy Authority
HL	Hamming Loss
I&C	Instrumentation and Control
IAEA	International Atomic Energy Agency
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
INCOSE	International Council on Systems Engineering
ISO	International Organization for Standardization
k-NN	K-Nearest Neighbor
LSTM	Long Short-Term Memory
LSTMs	Long Short-Term Memory Networks
ML	Machine Learning
MLP	Multilayer Perceptron
NLP	Natural Language Processing
NOAK	Nth-of-a-Kind
NPP	Nuclear Power Plant
NRC	Nuclear Regulatory Commission (United States)
ONR	The Office for Nuclear Regulation (United Kingdom)
PLM	Product Lifecycle Management
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SE	Systems Engineering
SoI	Systems of Interest
SoS	System of Systems
STUK	Finnish Radiation and Nuclear Safety Authority (Finnish: Säteilyturvakeskus)
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TFIDF	Term Frequency-Inverse Document Frequency
UK	United Kingdom
V&V	Verification and Validation
XGBoost	Extended Gradient Boosting
YVL	Regulatory Guides on Nuclear Safety (Finnish: Ydinturvallisuusohjeet)

List of Figures

List of Tables

1 Introduction

In the last four decades, nuclear power projects have become time-consuming and costs have globally increased partly due to increasingly complicated requirements. These requirements have affected the nuclear power industry rendering projects complex, therefore requiring large quantities of resources for the management of projects beginning from the licensing and design stages to the decommissioning phase (World Nuclear Association, 2013; IAEA, 2016; Schneider and Froggatt, 2018).

Requirements analysis is the most important task after discovering the initial set of requirements which occurs at the beginning of the project. At this stage, requirements should be analyzed as precisely as possible and potential conflicts found (Sommerville and Sawyer, 1997). In the licensing process of a nuclear power plant, not only will the requirements set by the national regulatory authority be met, but also any other applicable stakeholder requirements. They include, for instance, international standards, which might total tens of thousands of requirements (IAEA, 2010). However, this challenge is not nuclear industry specific since it also involves other safety critical systems. This subject has been widely investigated, and one of the overarching factors is that they all are heavily regulated industries (Goddard, 1996; Hatcliff *et al.*, 2014; Martins and Gorschek, 2016).

Effectively analyzing a vast amount of requirements necessitates the utilization of a machine, because individuals' ability to decide in face of such a number of options is extremely ineffective (Eysenck and Keane, 2010). Furthermore, studies reveal that the time range for a human being capable of sustaining attention on a specific matter is very limited (Lamba *et al.*, 2014; Bradbury, 2016). In contrast, a computer can constantly retain the same efficiency, the processing power being tremendous in comparison to human brains (Fischler and Firschein, 1987; Anusuya and Katti, 2010).

The importance of precisely analyzed requirements is especially emphasized as we consider the cost of change and nuclear safety during the project life cycle. It is well known that the more mature the project, the more it costs to fix errors (Boehm, 1981; INCOSE, 2015; NASA, 2016). The growth factor of the cost increases enormously when progressing through the life cycle (Haskins *et al.*, 2004). Furthermore, nuclear safety is better facilitated when the safety requirements are correctly evaluated. Therefore, an agile and correct analyzing method should be adapted in order to rationalize the whole nuclear power plant (NPP) life cycle, specifically the engineering and licensing processes.

This Master's thesis develops an artificial intelligence (AI) algorithm related to nuclear power industry in cooperation with an AI-company called Selko Technology Oy. The algorithm would accelerate the requirements analysis, and correctly obtain the requirements specification immediately at the beginning of the project. The efficient analysis process is essential for saving time (costs) as well as increasing nuclear safety by quickly and precisely applying suitable prerequisites to each design task. Generally, the wider intention is to investigate the applicability of AI in classifying requirements and rationalize the systems engineering process of which commencement is the requirements analysis. Because the requirements are written in natural language, a natural language processing (NLP) application and deep supervised learning are utilized as a part of the model for enabling the recognition and categorization of the requirements.

1.1 Background

In the initial stage of designing a nuclear power plant or a related safety critical system, the requirements and configuration management involve special consideration. The Regulatory Guides on nuclear safety (YVL Guides) issued by the Finnish Radiation and Nuclear Safety Authority (STUK) create a regulatory basis for all designs and their methods related to nuclear safety. The guides demand that the requirements be traceable and their fulfillment verified (STUK, 2013a). As authorities' requirements and the explication of these requirements become stricter, the importance of thorough design becomes more apparent once the licensing and engineering projects have started. However, categorizing, tracing and verifying the requirements can be extremely complicated due to the sheer volume of requirements, thus impeding a demonstration of compliance. Therefore, further research into requirements and configuration management has experienced an increased surge during the last few years.

One of the bodies in charge of internationally regulating nuclear industry companies is the International Atomic Energy Agency (IAEA), which establishes or adopts standards and main principles of safety for the protection of health and minimization of danger to life and property (IAEA, 1989). However, there is still an international lack of harmonization in the nuclear safety regulation, despite basic principles being similar worldwide as well as consistent with the policies of IAEA, and international rules being developed to improve the way of writing high-quality requirements. The requirements are expressed in different ways and each authority highlights slightly different demands, partly because they have idiosyncratically applied the principles (IAEA, 2006a, 2006b; Mavin *et al.*, 2009; MIT, 2018).

Each national regulatory authority is required to decide on actions needed to achieve compliance with national laws and regulations (IAEA, 2006b). Due to the legislative differences and the emerging state of international standards, a supplier has to be adaptable when executing any specific set of national or international safety requirements for its plant design; that is, a licensing process always includes adapting a country specific regulation (Fortum, 2018). Therefore, the consistency of the proposed design criteria must be assessed according to the national requirements and only considering the IAEA Safety Standards. Identifying new ways of adopting country specific requirements is considered one way of reducing unit costs. Usually, there are not only the nuclear and radiation safety requirements to be fulfilled, but also stakeholder requirements, which have to be considered in the licensing of a nuclear power plant. The stakeholder requirements, such as power production, startup, shutdown, maintenance and refueling of the plant compose adjunct demands (IAEA, 2000).

IAEA states that meeting the licensing requirements throughout the life cycle is as important as being able to adequately demonstrate the compliance to stakeholders. In addition to these requirements, the codes and standards define the applicable rules for components. Thus, there is a large variety of requirements, which may change in the various life cycle stages illustrated in Figure 1. At the beginning of the project, the definition of the project requirements, particularly the applicable codes, standards and regulations are considered to outline the suitable scope. Based on them, the requirements analysis, categorization and elaboration are performed prior to initiating designing. When new demands are elaborated, considering traceability of the requirements is essential to finally demonstrate the fulfillment of the applicable requirements (IAEA, 2010, 2012; STUK, 2013a).

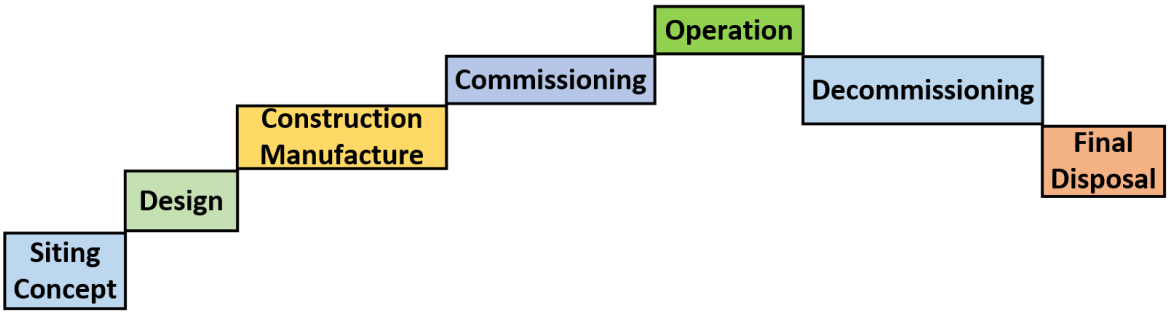


Figure 1 Stages in the lifetime of a nuclear installation (Adapted by the author from IAEA, 2010; INCOSE, 2015; Alanen and Salminen, 2016)

In other words, to systematically manage requirements of the nuclear power plant lifecycle, they should be assigned to relevant products and processes. Currently, the assignment is manually and gradually performed. Following the raw categorization, each label is also validated to adhere to the categorization, turning the requirements allocation into a time-consuming process. The World Nuclear Association (2013) specifies causes for the delay of the nuclear power projects, such as an initial application lacking quality or being incomplete, and requirements changing during the licensing or construction process. MIT especially highlights the effect of changes on plant design during construction, regardless of the reason for the change (MIT, 2018). To better facilitate the design, STUK requires that the design of systems important to safety shall be based on a life-cycle model (STUK, 2013a, p. 6).

A survey result reveals that there are huge variations in time needed for the preparation of the license application and for the licensing procedure. Although the wide range may be partly expounded by both various national regulations, and licensing systems and requirements (IAEA, 2012; World Nuclear Association, 2013), the import of whether a plant is first-of-a-kind (FOAK) or nth-of-a-kind (NOAK) is axiomatic. However, differences in regulatory regimes do effect on overall construction times and costs (Boldon and Sabharwall, 2014).

Table 1 describes a typical arrangement at the beginning of a licensing project. A customer has barely any expertise to participate in the commencement, and the authority should be independent and only focus on the supervision of the work concerning the nuclear safety. Consequently, this leads to the responsibility of the vendor supplying the plant. The supplier is required to analyze as well as categorize the requirements to be able to correctly allocate them, and as a consequence, perform possible design changes. Thus, the requirements should be precisely analyzed because the changes are carried out according to the analysis results, and the cost of changes increase along the life cycle as mentioned earlier and illustrated in more detail in Figure 2.

Table 1 Typical arrangement at the beginning of a licensing project

Party	**Special Characteristics**
Customer	- Presumes low costs - Imposes plenty of requirements
Authority	- Establishes nebulous requirements
Supplier	- Provides a standard product - Is willing to do business

The importance of the control of time, which should be one of the basic goals of all parties involved in a construction project, has been noted. It is mentioned that "the owner's goal is to shorten the time it takes for each phase of the project – from initial planning through construction execution" (Baker, 1991, p. 1). Given that time and money have a fixed relationship, time management should be carefully considered (Jung *et al.*, 2015).

Currently, the median average construction time for new reactors is 58 months (World Nuclear Association, 2018, p. 9), while time used in different licensing steps varies significantly worldwide as represented in Table 2. It should be emphasized that the fastest completion times have occurred in Japan: the first advanced boiling water reactor units were built in 37 and 39 months (IEA and NEA, 2015, p. 3). However, Japan's average construction time has been 47 months (World Nuclear Association, 2016, p. 21).

Table 2 Licensing steps and the related periods of time (Adapted by the author from World Nuclear Association, 2013)

Licensing Step	Time Range
Preparation of application	12 to 48 months
Construction license process	12 to 40 months
Operating license process	6 to 36 months

As mentioned earlier, the cost to fix errors increases along the life cycle. The following estimation illustrated in Figure 2 has been published by Haskins *et al.*, (2004). Fixing a requirements error during the requirements phase is assumed to cost 1 unit. Hence, the costs of similar events in the future can be compared to the reference event. It should be noted that the cost at the operations phase is estimated to be even more than 1500 units, but the simplified diagram is only plotted up to 100 units. Other studies have also been published sharing the same idea – the cost of the requirements error increases as the project matures (Rothman, 2002; Marasco, 2007).

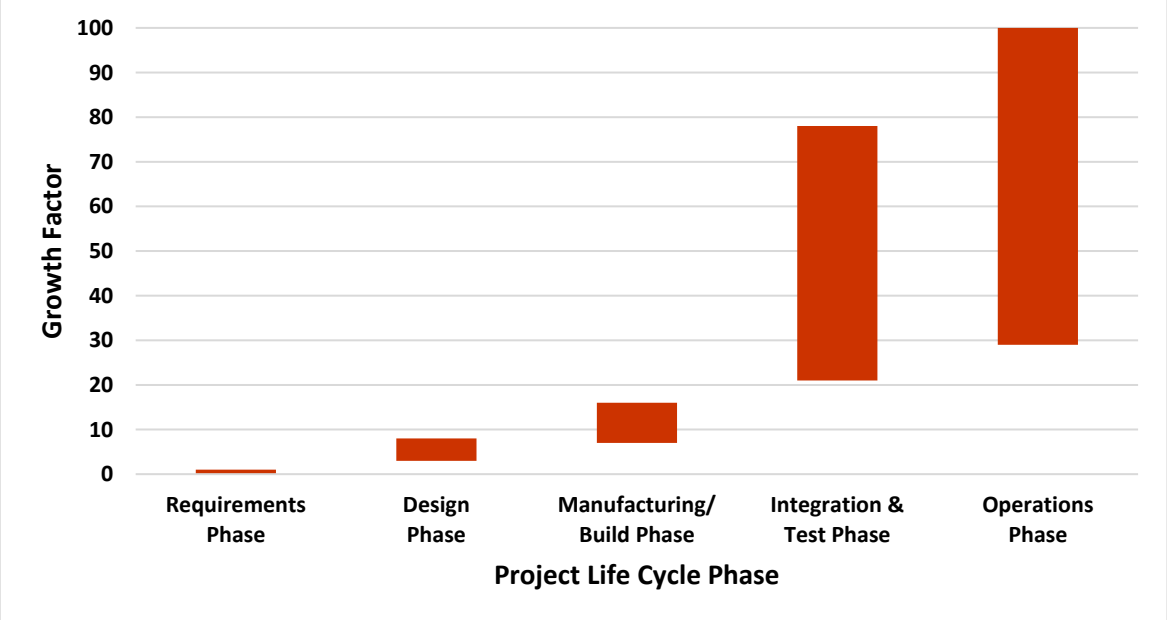


Figure 2 Cost of fixing a requirements error along the project life cycle (Derived by the author from Haskins *et al.* (2004)

Therefore, the requirements specification should be accurately performed directly after the commencement of the project. It has been recognized that the accuracy of the corrective categorization should increase with the help of artificial intelligence and the requirements categorization process become concurrently more efficient. When completion time and overall cost of the project are decreased, and operational performance as well as customer satisfaction are improved, the process is consistent with the lean approach. The approach aims towards increasing quality, decreasing price and reducing the duration of responds (Vujica Herzog and Tonchia, 2014; Smith and Thangarajoo, 2015).

The utilization of artificial intelligence and especially natural language processing in requirements analysis have been widely studied. A common objective connects each of the studies, utilizing artificial intelligence to improve processes, such as requirements classification and elicitation (Huyck and Abbas, 2000; Tamai and Anzai, 2018). In requirements analysis, an overarching challenge is that the requirements are usually implicit and descriptive. However, studies have already been performed in which the user only provides the requirements and the machine translates them into codes. It demonstrates the possibility of the utilization of AI. If performed correctly, it reduces costs and errors while changes are isolated in the requirements stage (Onowakpo and Ebbah, 2002; Sharma and Pandey, 2013).

1.2 Objectives and Scope

The objective of the research is to create a requirements categorization algorithm by utilizing a supervised learning method. The study also aims to improve and maintain professional expertise by increasing the understanding of systems engineering processes and artificial intelligence. The utilization of artificial intelligence (AI), specifically machine learning (ML), and natural language processing (NLP) would decrease the time used in the requirements analysis as well as increase the quality of the categorization. The requirements categorization algorithm is expected to be capable of more accurately classifying requirements than an expert and as such, improve the processes. The execution time of the algorithm should not be less than 80 requirements in a minute, and the accuracy should be better than a human judgement of 70 percent. The goal of the execution time and the judgement is based on the results of previous classification studies (Maggini, Rigutini, and Turchi, 2004; Khan *et al.*, 2018; Geirhos *et al.*, 2019).

This study is restricted to an experiment in which artificial intelligence and natural language processing methods are utilized based on supervised learning in requirements categorization aiming to investigate the prospect of the utilization of AI in the analysis of nuclear power plant requirements. Specifically, only certain YVL Guides and a requirement set issued by the Office for Nuclear Regulation (ONR) in the United Kingdom (UK) are used as well as specific categories. The initial objective is to test the ability of the algorithm to recognize different requirements, find similarities and label them according to a specific logic taught by specialists through the categorized learning data. It is highlighted that the focus of the thesis is on weak artificial intelligence which is emphasized by the narrowly defined problem (Al-Rifaie and Bishop, 2012; Miailhe and Hodes, 2017). The thesis relates to a broader intention to develop new tools and investigate prospects of utilizing AI technology especially in a specific licensing and safety engineering method developed by Fortum Oyj called ADLAS®. This proprietary method is Fortum's systems engineering approach. ADLAS® is further described and presented in Chapter 2.2.

However, the aim of the study is not to provide any detailed result of the algorithm's ability to be utilized in different cases of requirements engineering or even analysis, but to recognize the possibilities in which this or a similar algorithm could be utilized. Additionally, new targets for development are expected to be identified in this thesis. To support reaching the set targets, three main research questions and the related objectives are established and listed in Table 3.

Table 3 Research questions and the related objectives

Research Question	Objective
RQ1: What is the current stage of utilization of AI?	→ To evaluate the current possibilities of the utilization of AI in requirements analysis
RQ2: Where could AI and NLP be utilized along the life cycle?	→ To clarify the parts of the life cycle in which AI could be employed and determine the optimal methods
RQ3: What should be developed to better facilitate the utilization of AI?	→ To establish which issues should be considered to better enable the utilization of AI

The first research question (RQ1) aims to create a firm foundation of the current state of artificial intelligence and its ability to be utilized in requirements analysis. This is mainly achieved by developing an algorithm and testing it with a few different data sets. Before commencing the development of the model, a classified and high-quality data set is generated consuming the generality of the project time together with the development work of the algorithm; this is the primary focus of the thesis. The results will clarify issues already performable by the algorithm. The use of artificial intelligence in natural language processing has been hitherto limited; thus, the aim of the second research question (RQ2) is to clarify the parts of the life cycle in which NLP methods could be utilized. By analyzing the life cycle and activities related to systems engineering, further understanding will be gained of other possibilities for the utilization of NLP along the life cycle. The third question (RQ3) features enquiries of the most practical actions to be considered when the utilization of AI is further developed and better facilitated.

1.3 Execution of Study

The development project was performed by Fortum Power and Heat Oy (hereinafter referred to as Fortum) in close cooperation with Selko Technology Oy (Selko) which is a Finnish startup company focusing on developing AI algorithms for requirements engineering applications. The main responsibilities of the parties are presented in Table 4 below. The thesis worker was liable for the project, and collaborated with both internal experts and the external company. Generally, Fortum's responsibility was to provide a categorized training dataset as well as suitable testing datasets. Additionally, Fortum also validated the algorithm after the testing, whereas Selko was responsible for the development of the algorithm. In addition, Selko provided support in improving the understanding of artificial intelligence.

The execution of the study contains the following research phases: literature review, requirements categorization (training and test datasets), algorithm development, validation of the algorithm and conclusion of the applicability of AI technology. Figure 3 illustrates the execution of the study which includes each phase and its related actions in chronological order. The study began in the middle of August 2018, lasting until the middle of April 2019.

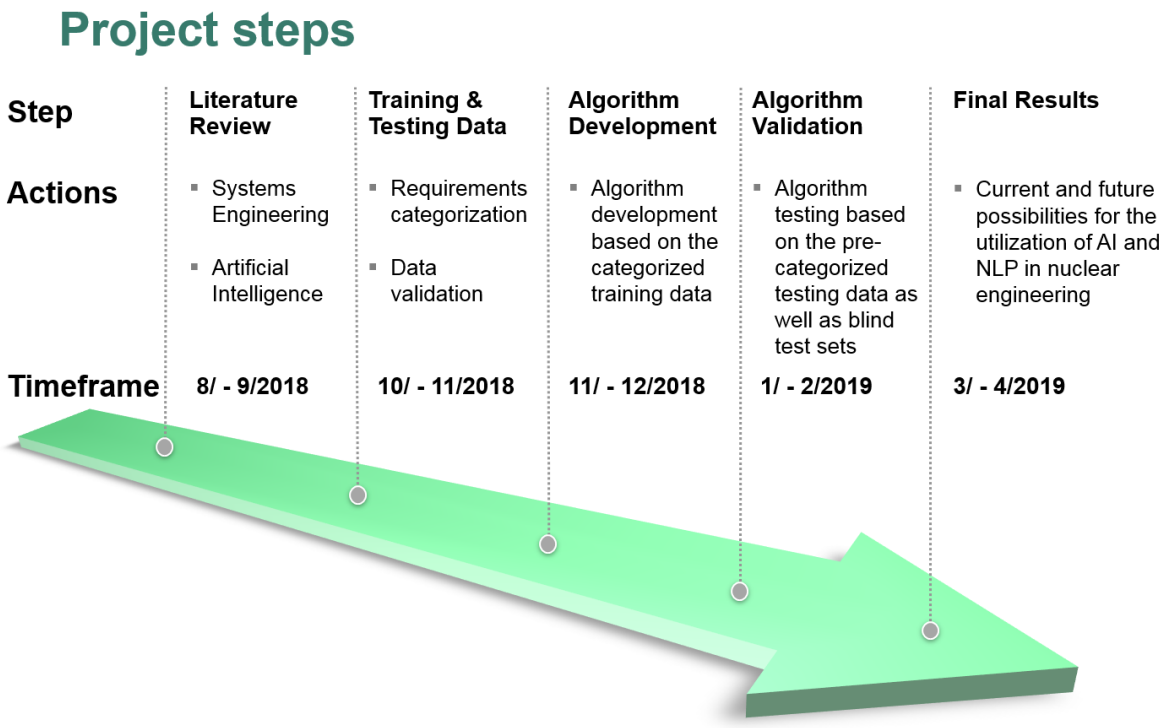


Figure 3 Timeline of the study

The project was initiated by planning the work and scope as well as reviewing earlier literature to have a common understanding of the two main subjects covering the study, systems engineering and artificial intelligence. In tandem with these actions, the YVL requirements were also categorized in which the aim is to have classified and high-quality training and testing data. Part of the initial dataset is then left for testing with the other part being used as a training set in the training phase of the algorithm usage. The classified and reviewed training dataset is used for teaching the functioning of the model. The testing phase is divided into three parts because different test sets are used to illuminate the current ability of the model and natural language processing in managing various regulatory environments. Specifically, the testing datasets include both the Finnish and the British requirements written in English. Finally, a conclusion is reached based on the results and observations from the literature and specialists.

In addition to the thesis worker responsible for the project, a core team of the development project on behalf of Fortum included the following experts: Technology Development Manager, Head of Nuclear Engineering, Design Engineer (Requirements and Configuration Management) and Nuclear Safety Design Manager. The team supported the completion of the project for which the thesis worker was responsible. In the core team, Selko was represented by Chief Executive Officer and two Data Scientists, specialized in neural network based classifiers. Furthermore, new development possibilities were discussed with Fortum experts located in Loviisa NPP (Finland) and Sweden.

The responsibilities defined and listed in Table 4 are generalized to clarify sharing of responsibility in general. In practice, there are many similar tasks since the testing phase is performed three times. Thus, specific datasets are separately collected, classified, and verified for each case. More precise descriptions of the tests are presented in Chapter 4.

Table 4 Responsibilities in the development project

Subject	Fortum	Selko
Definition of Objectives	X	
Theoretical Background and Literature Review	X	
Access and Understanding Data	X	
Data Collection	X	
Data Classification	X	
Data Verification	X	
Providing Data	X	
Creating Training and Test Datasets		X
Algorithm Development		X
Model Training		X
Model Testing		X
Model Scoring		X
Model Evaluation		X
Support for Validation Work		X
Verification of Results	X	
Model Validation	X	
Conclusions	X	

1.4 Structure of Thesis

Next, the structure of the thesis is briefly described including each chapter with both the related content and aim. The report consists of six main chapters. Chapter 1 provides the background for the thesis, illuminates the main objectives, provides limitations, and describes the execution as well as the structure of the thesis.

Chapter 2 presents the relevant theory in a form of literature review. Both main subjects covering the thesis are discussed, that is, systems engineering and artificial intelligence. The object is to provide the context of the thesis and to better understand the topics. However, interesting topics related to the main subjects are mentioned broadening the knowledge in these areas. Chapter 3 discusses the methods and data collection used in this thesis as well as trustworthiness of the study. It starts with representing the data collection which includes the exact documents and the amount of requirements utilized. The chapter demonstrates the overall way of developing the algorithm applicable also for this study.

The development project including each phase and the related results is thoroughly explained in Chapter 4. Most importantly, each finding is deliberated on, such as requirements classification and hierarchy as well as the model accuracies. In Chapter 5, relevant discussions are examined related to the present results and future possibilities. The discussions are based on interviews of the core team and other Fortum experts consulted during the project. Finally, Chapter 6 summarizes the study and the results accomplished through this project.

2 Literature Review

The two major disciplines employed in this study, namely systems engineering and artificial intelligence, are discussed in this chapter. Their theory will be reviewed focusing only on the relevant parts from the thesis point of view.

2.1 Systems Engineering

The aim of this subchapter is to emphasize the suitable theory of systems engineering which concerns designing and managing complex systems over the whole life cycle. The International Organization for Standardization (ISO) has established the ISO 15288 Standard *Systems and software engineering – System life cycle processes*, which defines Systems Engineering (SE) as an interdisciplinary approach enabling the realization of successful systems, more specifically "governing the total technical and managerial effort required to transform a set of stakeholder needs, expectations, and constraints into a solution and to support that solution throughout its life" (ISO/IEC/IEEE, 2015, p. 10). The standard continues to describe that SE integrates all the disciplines and specialty groups forming a structured development process proceeding from concept to production to operation.

In systems engineering, the concept of a system is defined as a "combination of interacting elements organized to achieve one or more stated purposes" (ISO/IEC/IEEE, 2015, p. 9). As mentioned in the same document, systems are defined by their functions, which are processes transforming resources from one state to another (ISO/IEC/IEEE, 2015). Figure 4 below represents a simplified hierarchy within a system in which the system acts as a boundary in order to achieve one or more stated purposes while the interacting system elements create the system. The schematic diagram shown in Figure 4 can already be called a system-of-interest (SOI). The related ISO Standard defines the concept of system-of-interest as a "system whose life cycle is under consideration […]" (ISO/IEC/IEEE, 2015, p. 9).

There may also be elements that are not part of the system but in which there is interaction. According to the INCOSE Systems Engineering Handbook (hereinafter referred to as the INCOSE Handbook), this collection of elements is called the *operating environment* or *context* and can include the users (or operators) of the system. The concept of a *system boundary* has been developed due to the system being visible from both the inside and outside. This boundary separates the system from its greater context, clearly defining the content forming part of the system. As the information traversing the subsystem boundary needs to be known, the same principle also appears in the interfaces of the subsystems (INCOSE, 2015). ISO 42010 Standard states that the environment of a system includes developmental, technological, political, legal, regulatory and ecological influences (ISO/IEC/IEEE, 2011b).

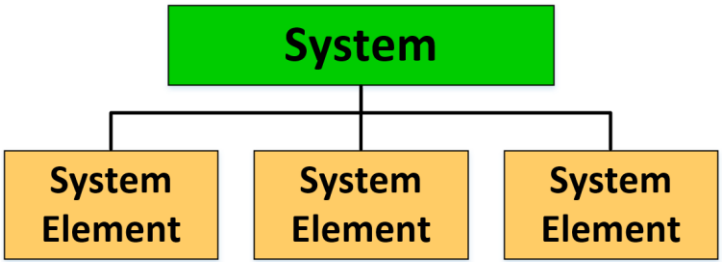


Figure 4 Hierarchy within a system (Adapted by the author from ISO/IEC/IEEE, 2015)

Each element can be either individual or at a much higher-level, acting more like a system itself. At any given level, a system can be formed by grouping the elements into distinct subsets. These subsets of elements are further subordinated to a higher-level system as illustrated in Figure 5 (INCOSE, 2015). ISO/IEC/IEEE 15288 (2015) describes the concept of a system hierarchy in Paragraph 5.2.2 as follows: "The system life cycle processes […] are described in relation to a system that is composed of a set of interacting system elements, each of which can be implemented to fulfill its respective specified requirements".

The ISO Standard 15288 also extends the idea of system-of-interest and represents a schematic diagram for more complex systems-of-interest as shown in Figure 5. As can be seen, the system-of-interest may consist of several different systems with some system elements even being considered to be new systems within systems. However, this is only a hierarchical relationship, while increasingly, systems are only partly hierarchical. Currently, networks and other distributed systems are examples of the concept of a system of systems (SoS), which is an SOI whose elements are managed and/or operated independently (ISO/IEC/IEEE, 2015).

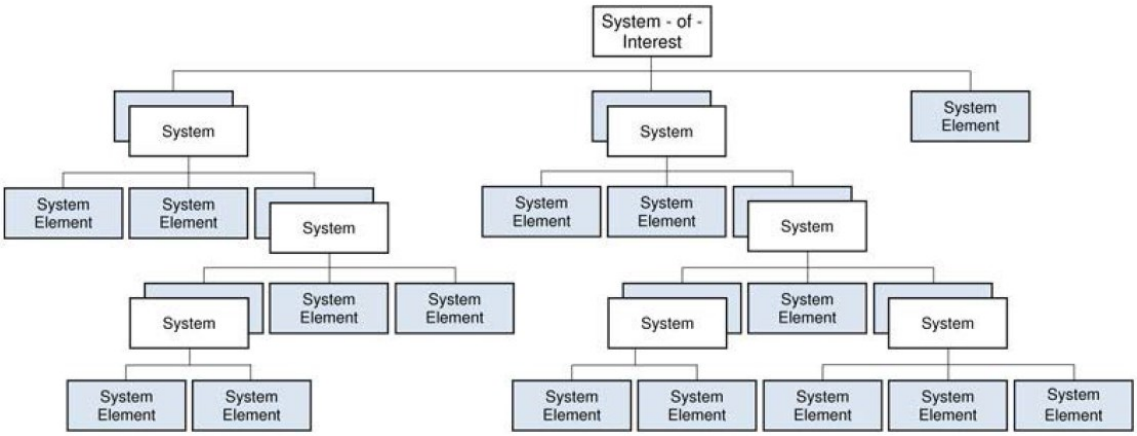


Figure 5 System-of-interest structure (ISO/IEC/IEEE, 2015)

To facilitate the life cycle activities of the SOI, enabling systems have been defined. They provide services needed by the SOI during any life cycle stage, although they are not directly part of the operational environment. The INCOSE Handbook provides examples of enabling systems, such as collaboration development systems, production systems, and logistics support systems. The enabling system may interact in conjunction with the SOI, or the SOI receives the desired services once needed (INCOSE, 2015). The SOI may simultaneously interact with various enabling systems while also interacting with systems comprising the operational environment (ISO/IEC/IEEE, 2015).

ISO 15288 Standard states that every system has a life cycle which consists of the conceptualization of a need of the system, its realization, utilization, evolution and disposal. In the context of a project, the life cycle includes the phases connecting the commencement of the project to its end (American National Standards Institute, 2004). From this point of view, both systems and projects can be seen as sharing similar characteristics. By way of processes used for execution of these actions, people performing and managing actions in organizations enable a system to progress through its life cycle. As system features, such as nature, purpose, use, and prevailing circumstances affect life cycles, they must be considered when planning and executing the system life cycle (ISO/IEC/IEEE, 2015).

2.1.1 System Life Cycle

ISO 15288 Standard states that every system has a life cycle which "can be described using an abstract functional model that represents the conceptualization of a need for the system, its realization, utilization, evolution and disposal" (ISO/IEC/IEEE, 2015, p. 14). There are many ways to determine a life cycle depending on the nature, purpose, use, and prevailing circumstances of the system. During planning and execution of the system life cycle, each stage is considered due to a distinct purpose and contribution to the whole life cycle (ISO/IEC/IEEE, 2015).

ISO 24748 Standard *Systems and Software Engineering – Life Cycle Management* lists typical life cycle stages: concept, development, production, utilization, support and retirement (ISO/IEC/IEEE, 2016). Product lifecycle management (PLM) defines life cycle stages particularly from a product point of view. Saaksvuori and Immonen (2008, p. 3) describes PLM as "a systematic, controlled concept for managing and developing products and product related information" offering "management and control of the product process and the order-delivery process", otherwise known as product development, productizing and product marketing (Saaksvuori and Immonen, 2008).

The life cycle stages identified by ISO 24748 and the related purposes are presented below in Table 5. Generally, the stages are sequential, but overlaps may exist. In contrast, *Utilization* and *Support* stages run in parallel during the operational life of the system-of-interest as Gray *et al.* (2017) state. As ISO 15288 specifies, each stage has a distinct purpose and contribution to the whole life cycle. The major life cycle periods are represented by the stages concerning "the state of the system description or the system itself" (ISO/IEC/IEEE, 2015, p. 14). The major progress and achievement milestones of the system are expressed by the stages which also generate the primary decision gates of the life cycle.

Table 5 Life cycle stages in a purpose-driven life cycle model (ISO/IEC/IEEE, 2016)

Life Cycle Stages	Purpose	Decision Gates
Concept	- Identify stakeholders' needs - Explore concepts - Propose viable solutions	Decision Options: - Execute next stage - Continue this stage - Return to a preceding stage - Put a hold on project activity - Terminate project
Development	- Refine system requirements - Create solution description - Build system - Verify and validate system	
Production	- Produce systems - Inspect and verify	
Utilization	- Operate system to satisfy users' needs	
Support	- Provide sustained system capability	
Retirement	- Store, archive or dispose of system	

2.1.2 System Life Cycle Processes

ISO 15288 defines four process groups, each of them including specific activities to be performed during the life cycle of a system, if necessary. It should be noted that performing a life cycle may not only be limited to the recognized processes but also any other processes may be utilized if considered useful (ISO/IEC/IEEE, 2015). The four process groups and the related processes are represented in Figure 6.

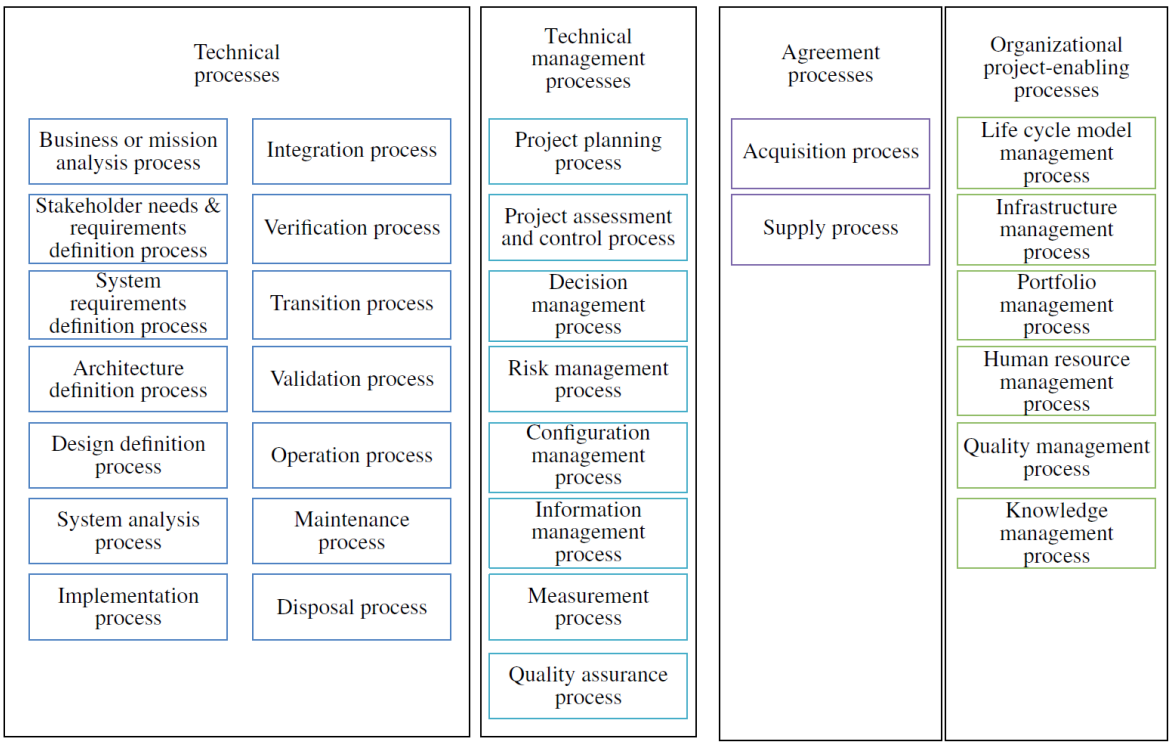


Figure 6 System life cycle processes defined by ISO 15288. This figure has been captured from the INCOSE Systems Engineering Handbook, originally being excerpted from ISO/IEC/IEEE 15288:2015, Figure 4 on page 16.

Technical actions are performed by technical processes throughout the life cycle. As stakeholders state their needs, they have to be considered and fulfilled in a product or service. This is ensured by using technical processes in the transformation. In addition, technical processes are implemented to establish and use a system. The processes can be utilized at any level in a hierarchy of the system structure and at any stage in the life cycle (ISO/IEC/IEEE, 2015). Product engineering is defined to involve "the technical processes to define, design, and construct or assemble a product" (ISO/IEC/IEEE, 2010, p. 273).

Technical management is stated to be "the application of technical and administrative resources to plan, organize, and control engineering functions" (ISO/IEC/IEEE, 2010, p. 366). Technical management processes manage the resources and assets of individual projects allocated by organization management. Thereafter, they are applied to fulfill the agreements into which the organization(s) enter. The technical effort of projects, especially planning of cost, timescales and achievements, are dependent on these processes. ISO 15288 amplifies their usage as follows: "to establish and perform technical plans for the project, manage information across the technical tasks through to completion, and to aid in the decision-making process" (ISO/IEC/IEEE, 2015).

Agreement processes are described as processes utilizing agreements for acquiring and supplying products or services; that is, conducting business with a supplier and agreeing that something is delivered to the acquirer. Since organizations are producers and users of systems, there is always at least one acting as an acquirer tasking another (acting as a supplier) for products or services. Organizations can simultaneously or successively act as both acquirers and suppliers of systems (ISO/IEC/IEEE, 2015). The acquirer establishes an agreement with the supplier and manage supplier performance by using the acquisition process (Electronic Industries Alliance, 1999). IAEA states that "the licensing process may also include agreements and commitments made between the regulatory body and the applicant" (IAEA, 2010, p. 5). For this reason, the licensing process may also be seen as an agreement process.

The outcomes of the business processes of the organization affect a project conducted in the particular context. The essential resources are provided by organizational project-enabling processes facilitating "the project to meet the needs and expectations of the organization's interested parties" (ISO/IEC/IEEE, 2015, p. 17). For instance, employees are needed to manage the project which may require certain facilities. The organization's capability to acquire and supply products or services at each project phase is partly ensured by these processes. They are not adequate to operate a business but "state the minimum set of dependencies that the project places upon the organization" (ISO/IEC/IEEE, 2015, p. 14).

2.1.3 System Life Cycle Models

Generally, systems may be defined as a sequential, single pass through the processes. However, an exchange of valuable information and insight should be enabled to effectively and efficiently meet the mission or business needs. It is essential to ensure that the information flows in every direction between the processes, thus, better facilitating the "incorporation of learning from further analysis and process application" (INCOSE, 2015, p. 32).

Life cycle approaches that attempt to facilitate the exchange of information have been recognized, such as the Waterfall (Royce, 1970), Spiral (Boehm, 1988) and Vee (Forsberg and Mooz, 1991). They can be used to define the beginning, ending and appropriate process activities (INCOSE, 2015). Next, two important life cycle models relevant to the thesis are presented, namely the Vee Model (also referred to as V-Model) and the Waterfall Model.

2.1.3.1 Waterfall Model

Originally in 1970, Winston Royce proposed the use of a waterfall model when managing large software projects. The model, originally aimed at software manufacturing, includes three important parts to improve the project management. Especially, it is stated that in a complex development project, there shall be several upper-level steps. This is illustrated in Figure 7 (Royce, 1970).

With respect to software industry, analysis and coding are the most important phases when developing a large-scale program, but not enough alone to manage and control the development process. Therefore, supplement steps are introduced: system requirements definition, software requirements definition, program design and testing. The left-side approach of Figure 7 is dependent upon the iterative interaction between the various phases. In practice, in an implementation of the approach, possible failures, such as timing, storage and input/output transfers are experienced only in the testing phase at the end of the development cycle. This means the product improvement returns to either the software requirements (modify the requirements) or the program design (substantial change has to be made) being a time-consuming and expensive process. Inserting a preliminary program design, as shown in the right-side approach of Figure 7, enhances abilities to observe possible failures early enough to be able to easily modify the design. According to Royce, to further improve the overall development process, documentation should also be current and complete. Hence, the preliminary program design phase includes documenting system overview, designing data base and processors, allocating subroutine storage as well as execution time, and describing operating procedures. In addition to the required documentation, the stage includes preliminary design, analysis, program design, coding, testing and usage. That implies the job is performed twice (Royce, 1970).

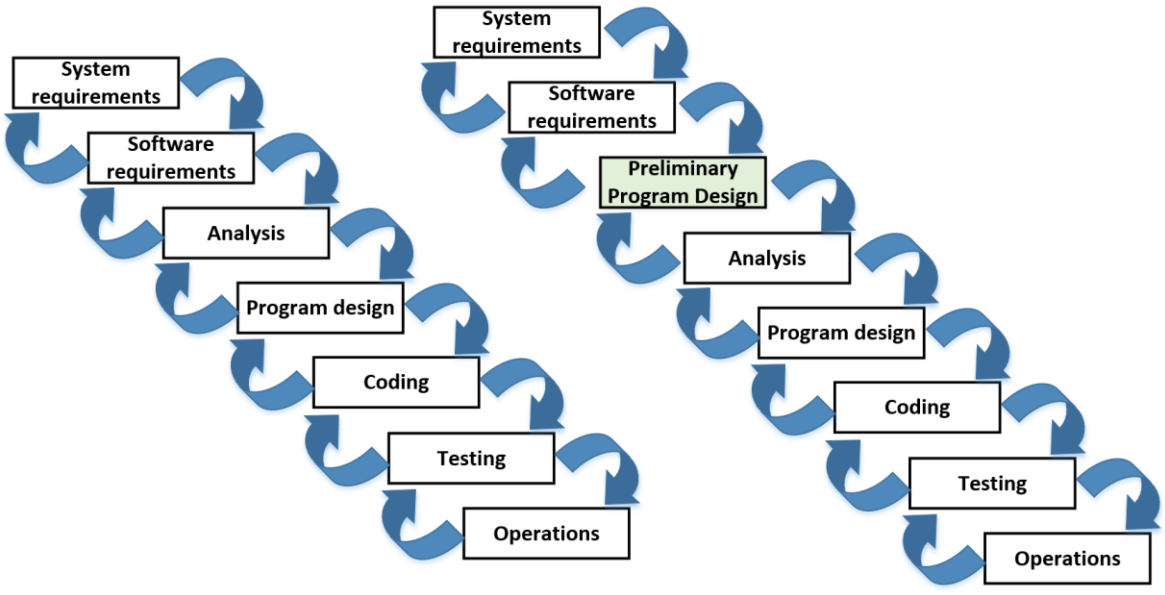


Figure 7 The large-scale system approach with and without a preliminary program design (Adapted by the author from Royce, 1970)

While the original Waterfall model is based on the iterative interactions, an incremental model has been recognized in software development resulting in a "multi-waterfall" cycle. Multiple development cycles occur in the incremental model as designing, implementing and testing are incrementally performed. The product is divided into builds to separately create and test sections of the project. By defining several smaller baselines facilitates the finding of errors in user requirements. This is because of soliciting feedback for each stage, and testing the current version instantly it has been finished. The flexibility in changing scope and requirements is also considered one of the advantages of this approach (Mandal, Kandar, and Ray, 2011; Singh, Thakur, and Chaudhary, 2015).

Lutz and Huitt (2003, p. 2) describes that the interaction of new information with stored information is usually demonstrated with a bottom-up or top-down system, or a combination of the two. The latter approach is also known as Both-Ends-against-the-Middle (BEATM) design. The primitive implementation steps to develop a large computer program presented by Winston Royce (1970) exemplify a top-down (also called as an allocation and flow-down) design which begins with a view of an important problem that needs a solution. The design focuses on high-level requirements which are further decomposed into lower and lower-level structures and specifications. Finally, the physical implementation layer, existing also in the V-model and ADLAS® methodology, is reached (INCOSE, 2015; Keyes, 2015). The bottom-up design focuses on the potential of available real-world physical technology, implementing solutions to which the technology is most suited. Both the bottom-up and top-down designs have specific questions to which they attempt to respond, respectively. These questions are listed below. Keyes (2015, p. 13) specifies that the BEATM design immediately focuses at both ends of the design process flow: "a top down view of the solution requirements, and a bottom-up view of the available technology that may offer promise of an efficient solution".

- "What can we most efficiently do with this technology?" (Bottom-up)
- "What is the most valuable thing to do?" (Top-down)

From a psychology point of view, a bottom-up system sees new information "as an initiator which the brain attempts to match with existing concepts to break down characteristics or defining attributes" (Lutz and Huitt, 2003, p. 2). Conversely, the existing information is stated to be utilized as the initiator in a top-down system (Lutz and Huitt, 2003). That is, initial knowledge essential to form user requirements might exist, which has to be considered in order to adequately define the requirements at the beginning of the project (Forsberg and Mooz, 1991, p. 6).

As discussed, the BEATM design process simultaneously initiates from both ends attempting to find an optimum merging. It has been recognized that some of the successive exploitations of the two separate processes are due to an intuitive, yet unconscious use of the BEATM methodology (Keyes, 2015). The Waterfall Model established for software development is also applied to the ADLAS® methodology with the Vee Model. The Vee Model will be discussed in the following subchapter, whereas the ADLAS® is later described in Chapter 2.2.

2.1.3.2 V-Model

In 1991, Forsberg and Mooz visualized the technical aspect of the project cycle as a "Vee", in which there are user needs on the upper left and a user-validated system on the upper right, the cycle starting and ending respectively (Forsberg and Mooz, 1991). The model was further introduced by Forsberg, Mooz, and Cotterman (2005) as they described the model in more detail. The architecture of the V-Model is shown below in Figure 8. The word "architecture" is generally used to describe the way of which the subsystems join together to form the system (Department of Defense, 2001, p. 6). According to SFS-EN 61508-4 Standard, an architecture is defined as a "specific configuration of hardware and software elements in a system", indicating that the meaning of architecture is dependent on the definition of the system-of-interest (SFS, 2010).

The left side of the Vee reflects the well-established Waterfall Model for the project cycle. The system definition (top-down branch) is conducted by successive levels of decomposition, each of them corresponding to the physical architecture of systems and system elements. There is no limit in the amount of levels in the decomposition. For instance, the INCOSE Handbook defines seven decomposition levels. In contrast, the integration (bottom-up branch) forms the opposite way in which each level is separately and sequentially composed starting from the bottom (Forsberg and Mooz, 1991; Forsberg *et al.*, 2005).

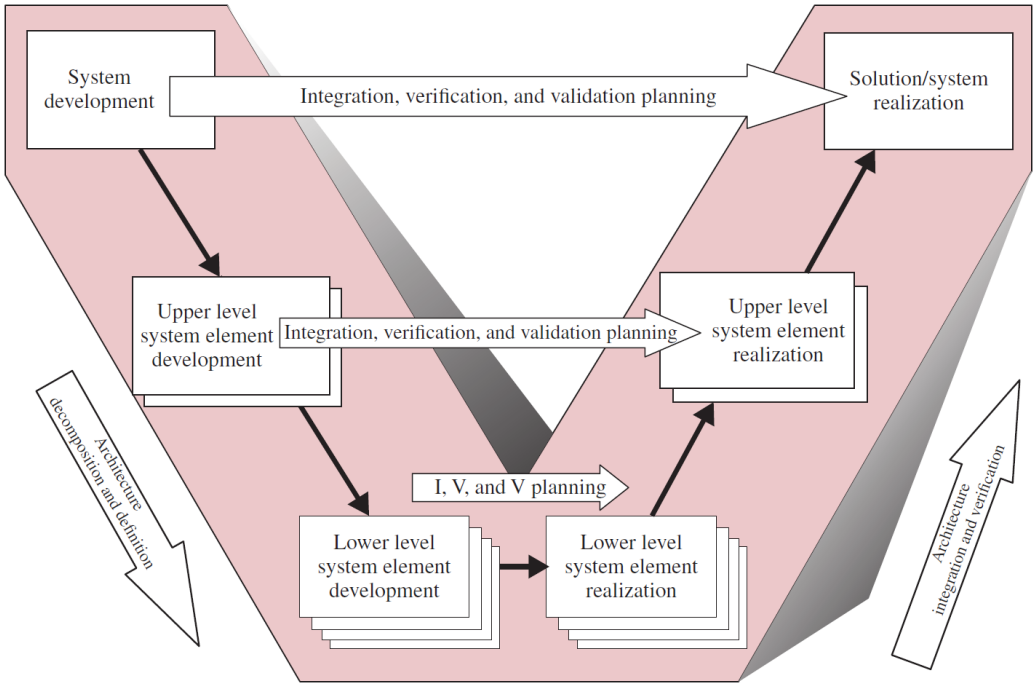


Figure 8 Vee Model (Captured from INCOSE, 2015, originally being excerpted from Forsberg *et al.*, 2005)

Forsberg *et al.* (2005) emphasize that the activities on the left and right sides of the Vee are connected with each other. The verification and validation methods to be used on the right are already determined in the definition stage. As Figure 8 illustrates, there is a direct correlation between activities. Both the defined system and systems elements belong to system definition side including requirements and design characteristics. The design is verified against the realized system and systems elements. Verified products form a realized system which completes the final product in the system realization branch (INCOSE, 2015).

2.1.4 Requirements Analysis

In successful projects, the needs and requirements of the stakeholders have to be met throughout the life cycle. System's development is governed by the stakeholder requirements, thus, being used in further definition or clarification of the scope of the development project. System definition is based on the systems requirements which are established from the defined stakeholder requirements. A complete but minimum set of requirements should be defined due to a cost of each requirement (INCOSE, 2015). Requirement can be defined as "a condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed documents" (American National Standards Institute, 2004, p. 371). The same document also highlights that the quantified and documented needs, wants, and expectations of any stakeholder can be considered requirements (American National Standards Institute, 2004).

ISO 15288 Standard describes the *Stakeholder Needs and Requirements Definition Process* which aims to define the requirements for a system necessary in providing the needed services. In addition to the identification of stakeholders or stakeholder classes as well as their needs, expectations and desires, the process "analyzes and transforms these into a common set of stakeholder requirements that express the intended interaction the system will have with its operational environment and that are the reference against which each resulting operational service is validated" (ISO/IEC/IEEE, 2015, p. 51). Similarly, system requirements are defined and analyzed as a part of *System Requirements Definition Process* of which purpose is to transform prerequisites of the stakeholder into a technical view of a solution satisfying the operational needs of the user (ISO/IEC/IEEE, 2015). These processes constitute requirements analysis process, also referred to requirements engineering, being a part of requirements management, which is a subset of systems engineering. The requirements analysis process aims to "provide an understanding of the interactions between the various functions and to obtain a balanced set of requirements based on user objectives", according to the INCOSE Handbook (INCOSE, 2015, p. 60). As seen in Figure 9, requirements analysis is the first stage in the systems engineering process (ISO/IEC/IEEE, 2011a; INCOSE, 2015).

Requirements analysis, or requirements engineering, is defined to focus on "discovering, developing, tracing, analyzing, qualifying, communicating and managing requirements that define the system at successive levels of abstraction" (Hull, Jackson, and Dick, 2011, p. 8). Regarding to systems engineering, Figure 9 illustrates the importance of requirements analysis in systems engineering process. The process' primary purpose is stated to be transforming the requirements into designs. Other fundamental systems engineering activities include functional analysis and allocation as well as design synthesis. The activities which represent a perception of the second party, namely Department of Defense, are balanced by techniques and tools collectively called system analysis and control, which together with other activities refer to system life cycle processes introduced in Chapter 2.1.2 (Department of Defense, 2001; Hull *et al.*, 2011).

Nine characteristics of individual requirements are outlined in ISO 29148 Standard, four of which are influential from the perspective of the thesis and highlighted as follows: unambiguous, singular, traceable and verifiable. "Unambiguous" means that the requirement can easily be understood, and in only one way. "Singular" refers to the statement with only one requirement and without any conjunction. "Traceable" is based on the idea that "all

parent-child relationships for the requirement are identified in tracing such that the requirement traces to its source and implementation". Finally, "verifiable" indicates that it shall be feasible to justify the conformity of the system. To facilitate the employment of good requirements characteristics, requirement language criteria have been published in ISO 29148, stating that it is more important to know the needs for the system-of-interest rather than any design decisions for it. For this reason, neither vague and general terms nor ambiguous terms should be used in requirements (ISO/IEC/IEEE, 2011a, pp. 11–12). These characteristics of requirements are considered later when evaluating the research results.

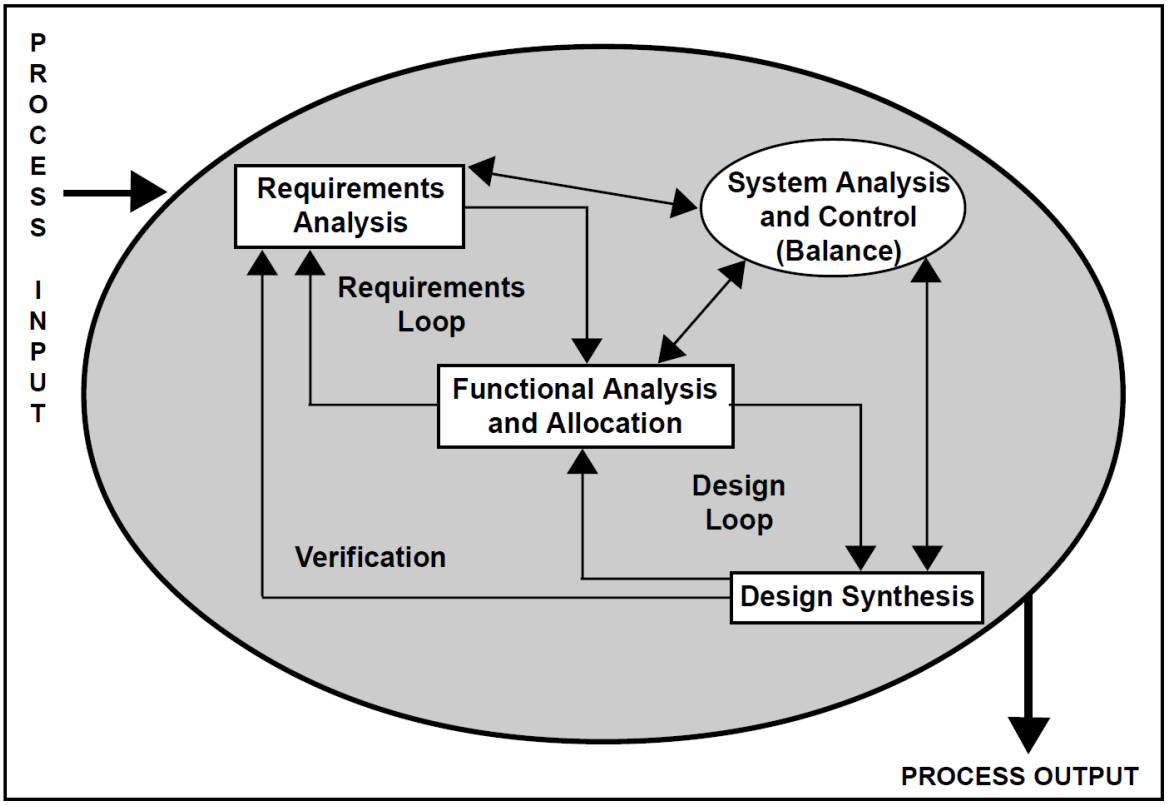


Figure 9 Systems Engineering Process (Department of Defense, 2001)

Descriptive attributes should be defined to facilitate understanding and management of the requirements. Definition of the attributes is performed to support requirements analysis as the attribute information should be associated with the requirements. ISO 29148 Standard describes important examples of requirements attributes, two of which are the most essential regarding to this research, namely identification and type. Each requirement should be uniquely identified by using a unique identifier (i.e., number, name tag, mnemonic) which assists in requirements tracing. The label shall permanently remain unchanged. Defining the type for each requirement facilitates the collection task of grouping requirements into determined categories for analysis and allocation, because the requirements include divergent intents and properties. From the list of examples of the requirements type attribute, two important requirements type attributes are emphasized; functional and process requirements (ISO/IEC/IEEE, 2011a). According to the standard, "functional requirements describe the system or system element functions or tasks to be performed", while process requirements "are stakeholder […] requirements imposed through the contract or statement of work" (ISO/IEC/IEEE, 2011a, p. 14). Similar types will be utilized in this study, but only the contents of the categories are differently specified.

Various development stages and the related testing phases were represented in Figure 8. To highlight the importance of requirements engineering at every stage of development, Figure 10 demonstrates how the defined requirements are exploited in testing as everything is tested with respect to requirements. Stakeholder requirements reflect the results according to which the product is validated, system requirements define the functions for the system(s), subsystem requirements aim to optimize the cost-benefits and finally, the component requirements are allocated to each component. Beginning from the component or product stage, testing proceeds stage by stage until it is confirmed that the product fulfills the set requirements, especially the stakeholder requirements. Again, both the top-down and bottom-up designs can be performed in eliciting requirements, and to manage changes, traceability and impact analysis may be utilized. Requirements tracing is important, since in the matter of changing the design of a product, the requirements reflecting that change have to be updated (Hull *et al.*, 2011; INCOSE, 2015).

According to Hull *et al.* (2011, p. 78), one of the key capabilities required for requirements is the "ability to elaborate a requirement in multiple ways by providing performance information, quantification, test criteria, rationale and comments". Elaboration consists of eliciting and analyzing requirements to profoundly understand the needs of the stakeholders to support the architecture definition and design definition processes. Leffingwell and Widrig (1999) introduced natural identification scheme for hierarchical requirements in which child-requirements are identified following parent requirement's identification. The parent-child relationship is viewed "as an amplification of the specificity expressed in a parent requirement" (Leffingwell and Widrig, 1999, p. 185). A uniquely identified item (child) is associated with the next higher-level of assembly having a hierarchical relationship to its parent. That is, the lower-level item of the parent-child relationship is indicated as the child, whereas the parent is the higher-level item of a parent-child relationship. In addition to elaboration, there are many other tasks to be performed, such as identification, classification, tracking status, tracing, placing in context and retrieving. Hence the importance of expressing and organizing requirements is highlighted to enhance the usability of requirements sets (Leffingwell and Widrig, 1999; Hull *et al.*, 2011; INCOSE, 2015).

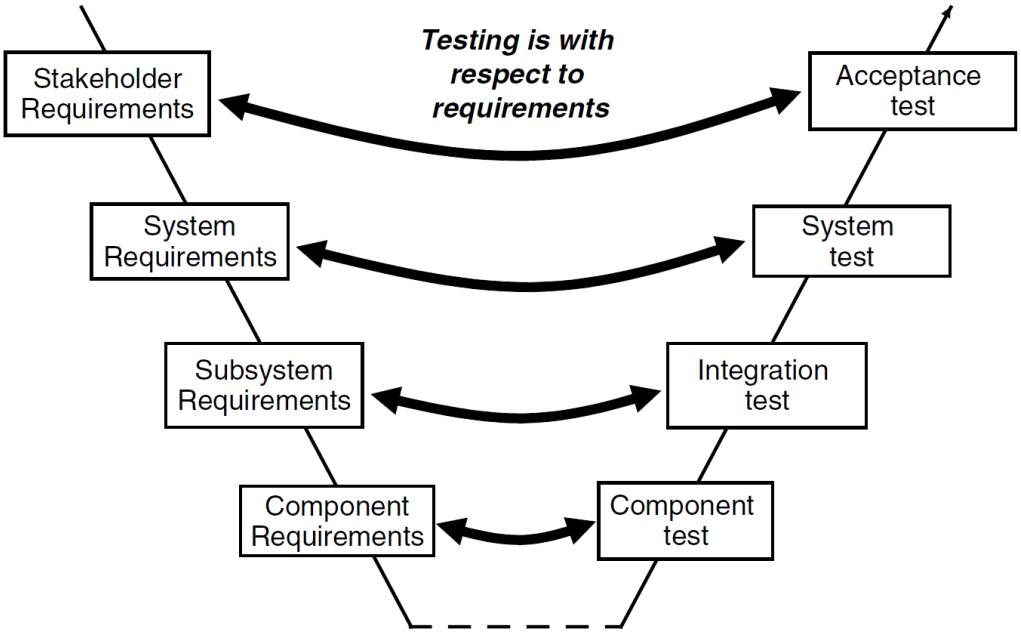


Figure 10 Requirements and the corresponding V&V activities in the V-Model (Hull *et al.*, 2011)

2.2 Advanced Licensing and Safety Engineering Method

Due to the complexity and high quality requirements of safety systems, the investment costs of traditional nuclear power plants have increased and the projects have suffered from budget and schedule overruns. Traditionally, the licensing documentation is complicated and descriptive, requirement engineering lacks high-level requirements or they are insufficient in system-level, and configuration management concentrates only on the physical configuration. Furthermore, requirements may change during plant lifetime. Therefore, a need for a requirement specific and hierarchical licensing and safety design method has been arisen, which would ensure the fulfillment of the requirements in an effective way (Nuutinen, Sipola, and Rantakaulio, 2016).

Usually, design requirements are not provided in licensing documentation. In addition, regulatory requirements are descriptively written requiring interpretation. However, there is no documentation clearly stating the interpretation of regulatory requirements during the design, leading to a situation in which the licensing documentation does not provide sufficient information for major changes in nuclear facilities. Moreover, the documentation does not provide such information in many new build projects even though transparency and traceability in the design process are required, such as in Finland (STUK, 2013a), Hungary (HAEA, 2015) and United Kingdom (ONR, 2016) (Nuutinen, Sipola, and Rantakaulio, 2017).

Therefore, Fortum has developed a new high-level safety engineering method called *Advanced Licensing and Safety Engineering Method*, ADLAS® (hereinafter referred to as ADLAS), to illuminate the licensing aspects and the safety features behind the licensing requirements. The method is a systematic and well-documented way of preparing the licensing documents. It has already been utilized in the licensing of nuclear facilities, and is being applied in several projects by Fortum. ADLAS provides a transparent safety engineering process and hierarchy for the requirements, elaborates them to the actual design requirements, and justifies the scope and extent of the plant and architecture-level V&V activities. In addition, it provides the basis and method for configuration management on plant and architecture-level as well as the system-level requirements from the safety design view point (Korhonen and Nuutinen, 2016; Nuutinen *et al.*, 2016).

As mentioned in Chapter 2.1, the ADLAS methodology utilizes the described systems engineering approaches as the large-scale system approach is combined with the V-model. Figure 11 shows schematically the way of which the BEATM approach is utilized in ADLAS. The advantage of combining the Waterfall Model and the BEATM approach is that the essential higher-level requirements can be recognized at the lower-level. As a result, they are considered and elaborated, for instance, due to a design constraint. On the other hand, the V-Model involves verification activities at each level. Hence, the requirements are hierarchical from top to down, essential lower-level requirements are already considered at the higher-level, and the configuration management method is also applicable for requirements and engineering data, improving intelligibility (Nuutinen *et al.*, 2016).

As Korhonen and Nuutinen (2016, p. 1) states, “to obtain safe operability in safety critical systems, components comprised in such systems may be associated with safety conditions”. Therefore, two key methodological aspects of ADLAS include categorizing high-level principles into a specific group called *Essential Safety Requirements*, and establishing *Task Categories* (Nuutinen *et al.*, 2016).

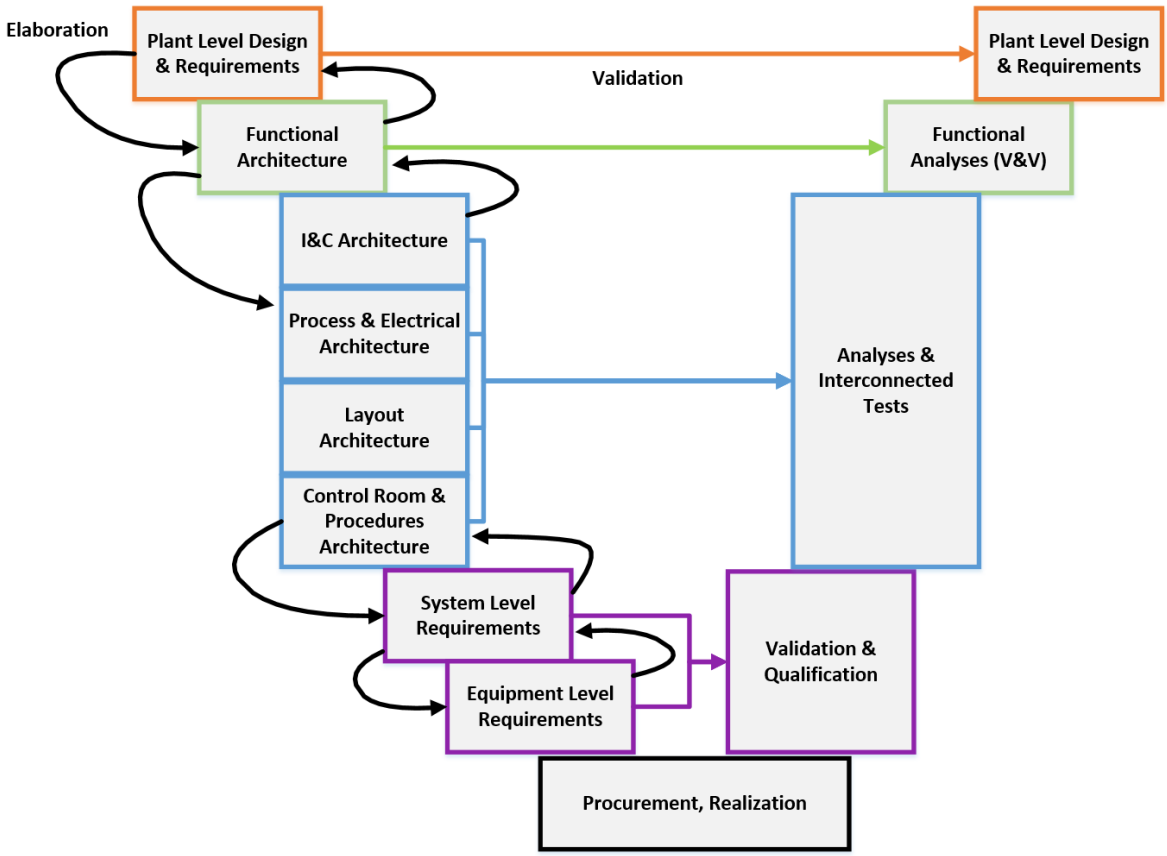


Figure 11 Both-Ends-against-the-Middle approach applied in ADLAS® (Adapted by the author from Nuutinen *et al.*, 2016)

Essential Safety Requirements include the characteristics described in ISO 29148 and partly mentioned in Chapter 2.1.4, whereas "Task Categories combine functional and non-functional safety requirements into a group of safety functions", according to Nuutinen *et al.* (2017, p. 5). That is, the task categories are functional entities or groups of functions sharing similar functional (end-state) and non-functional (i.e., redundancy or diversity) requirements. The category is associated with a functional requirement and design principle, and further with an architecture definition information element which is again associated with at least one system-level information element. There may be more than one functional requirement, design principle or information element in place. The formed system is then verified in a bottom-up manner to ensure the system is compliant with the defined design principles. Due to different demands of systems in various situations, the task categories set requirements for every technical discipline through the safety functions (Korhonen and Nuutinen, 2016).

Functional design lists the safety functions required for mitigation of initiating events. The functions concern three main safety objectives introduced by IAEA (2006, pp. 4–5). The aim is to define the actions the systems should perform instead of describing the quality of the functions. According to Nuutinen *et al.* (2017), Task Categories are assigned to the design basis categories so that the non-functional requirements of the safety functions can transparently be seen in the design document. The safety functions are also assigned to the recognized initiating events.

2.3 Artificial Intelligence

The development of artificial intelligence (AI) began already in 1950's. John McCarthy was the first introducing the term *artificial intelligence*, combining the fields of cybernetics, mathematics, algorithms and network theories. Initially, an attempt was to discover ways of generating machines to use language, form abstractions and concepts, solve various problems and improve themselves (McCarthy *et al.*, 1955). He defines the field as "the science and engineering of making intelligent machines, especially intelligent computer programs" (McCarthy, 2007, p. 2). Another, more specific definition of AI was given in 1993: "the study of the computations that make it possible to perceive, reason, and act" (Winston, 1993, p. 5).

As Gabriella Daróczy mentions in her article *Artificial Intelligence and Cognitive Psychology*, the main goal in developing AI is reaching human-level intelligence (Daróczy, 2010). Initially, the goal was to create something more clever than human. During the last centuries, it was noticed that it would be more beneficial to focus on designing applications possessing some intelligence, or more precisely, seem to be intelligent. As some AI methods have become well known, they are not even considered belonging to AI anymore. The AI methods can be presented, for instance, in the form of an algorithm or as a part of an architecture of a software system (Tyugu, 2007). This field has provided us many applications, such as self-driving cars, practical speech recognition, effective web search, analyses of brain scans and predictions of tumors, but especially an improved understanding of the human genome (Onowakpo and Ebbah, 2002; Crnkovic-Friis, 2018).

Due to its ambiguity, the term "intelligence" should be defined. It has been stated that it is closely related to knowledge and ability of knowledge handling (Tyugu, 2007). A study shows that there are a huge amount of different ways to define intelligence, and very importantly, there is still no standard definition of intelligence (Legg and Hutter, 2006). Below, there are three definitions of intelligence: one psychologist and two AI researcher definitions, respectively. Especially, the latter two definitions are suitable for this study.

- "We shall use the term 'intelligence' to mean the ability of an organism to solve new problems . . . " (Bingham, 1937)
- "Intelligence is the ability to process information properly in a complex environment. The criteria of properness are not predefined and hence not available beforehand. They are acquired as a result of the information processing." (Nakashima, 1999)
- "Intelligence means getting better over time." (Schank, 1980)

Artificial intelligence can support reaching engineering goals in solving real-world problems, optimizing jobs and activities, minimizing risks as well as helping to decide and exclude certain logical tasks performed by a human. However, the challenge is to solve tasks which people can easily perform but hardly formally describe. New opportunities which neither can realize alone, may be recognized as the abilities of intelligent people and computers are combined (Winston, 1993; Goodfellow, Bengio, and Courville, 2016; Crnkovic-Friis, 2018). Being an interdisciplinary branch of computer science, Kocaleva *et al.* (2016, p. 236) report that AI has connections to other sciences, such as neuroscience, philosophy, linguistics and psychology. Areas of specialization of artificial intelligence include games playing, expert systems, natural language, neural networks and robotics (Kocaleva *et al.*, 2016).

Learning programs may mostly be divided into experience or data-oriented categories. In the experience-oriented learning, programs learn humans' way of performing tasks by reasoning about new experiences in the light of practical knowledge. Therefore, the knowledge computer needs is unnecessary to be formally specified. On the contrary, practical programs capable of mining databases for exploitable regularities are aimed to be developed in the data-oriented learning. Initially, it was demonstrated that computers can work problems in integral calculus at the level of college freshmen, whereas currently, programs perform mathematical analyses at a much higher-level. To transform informal knowledge which is commonly subjective and intuitive, into a computer, is considered one of the key challenges in AI (Winston, 1993; Goodfellow *et al.*, 2016). Hence, AI systems are expected to have the ability to acquire their own knowledge by extracting patterns from raw data. This capability is known as *machine learning* (Goodfellow *et al.*, 2016).

Figure 12 presents the intersections of data and computer science as well as the related subfields of computer science. As emphasized, deep learning, machine learning and artificial intelligence belong both to the field of data science and computer science. Data science covering computer science and its subfields is defined as "the management and analysis of data sets, the extraction of useful information, and the understanding of the systems that produce the data" (Suthaharan, 2016, p. 1). The same system definition is considered as defined in Chapter 2.1. Therefore, an example of a single unit is a computer network composing of interconnecting subunits (computers). As complex systems generate large amount of unstructured and complex data which are hard to manage, process and analyze, a term *big data* is referred to. Mathematical models and algorithms are utilized to efficiently classify the data forming the field of machine learning (Suthaharan, 2016).

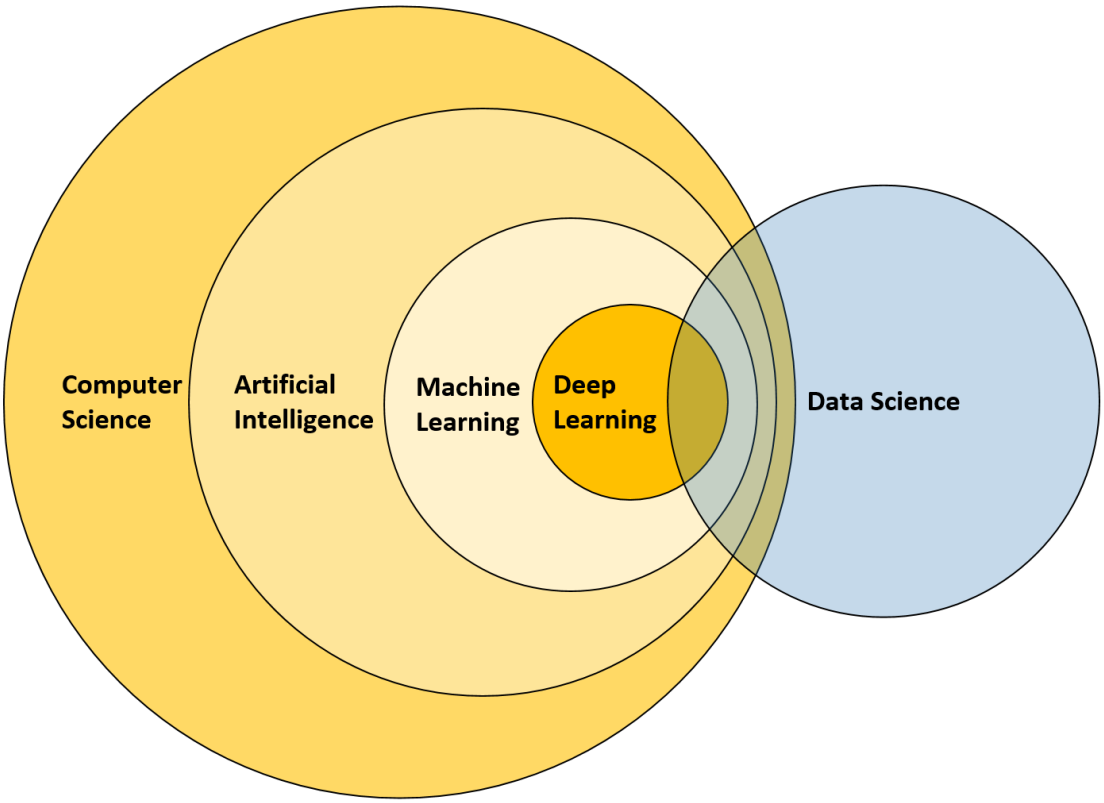


Figure 12 A Venn diagram representing the relationships between different fields (Derived by the author from Winston, 1993; Loukides, 2010; Goodfellow *et al.*, 2016; Crnkovic-Friis, 2018)

To ensure a common understanding on the specific terms relevant to the thesis, the corresponding definitions are explained in Table 6. The listed terms and their definitions are not unambiguous, but rather described considering the context. The labeled data are used for training the machine learning techniques while the model is tested and validated using unlabeled data. To be precise, the testing data have to also be labeled to perceive accuracies of the model when the results are compared to each other. However, labeling may occur prior to feeding the test data or after having the results. In the latter case, the results should not be seen in advance (Suthaharan, 2016).

Table 6 Data and computer science related terms and definitions relevant to the thesis (Adapted by the author from Goldberg, 2015; Goodfellow *et al.*, 2016)

Term	Definition
Data	Hidden digital facts which the monitoring system collects. In labeled data, the facts are not hidden, whereas unlabeled data consist of the hidden facts
Algorithm	Detailed description or precept about the way a task or process will be conducted
Knowledge	Learned information acquired from the data, e.g., detection of patterns or classification of the varieties of patterns in the data
Parameter	Value that controls the behavior of the system
Feature	Concrete, linguistic input, such as a word or a part-of-speech tag
Token	A word, character, or even byte. Tokens are always discrete entities.

Winston (1993, pp. 43–44) lists questions about the essential knowledge when approaching a new class of problems. The questions relate to the nature of knowledge involved, the way of representing that, the amount of knowledge required, and the exact knowledge needed. The important knowledge may concern the description of concrete or abstract objects, or alternatively, it is about a problem-solving method. The representation possibilities include a semantic-net framework and a collection of procedures illustrated by Winston (1993). Once the characteristics of knowledge are recognized, the quantity of items to be known should be clarified to consider the demand for sensible resource allocation as well as to build courage by knowing the size of a problem. However, there is a risk to overestimate a complicated task to be unimaginably complicated even though the task may be performed by a human using a little knowledge (Winston, 1993).

The same principles apply regardless of the destination of the extracted knowledge, that is, computers or humans. Specific situations and situation pairs looking identical but that are differently handled, are asked to learn general knowledge and vocabulary, respectively (Winston, 1993). This leads to the capabilities of machine learning and applications within that field. Basics of the relevant applications, specifically, artificial neural networks and natural language processing, are presented in the following chapters. These subjects form the core of the developed algorithm as can be observed.

Three questions have been set to determine whether a research work in AI is successful. On the other hand, two questions are defined to discover whether an application of artificial intelligence is successful (Winston, 1993, pp. 13–14):

- “Does the application solve a real problem?”
- “Does the application open up a new opportunity?”

2.3.1 Machine Learning

As discussed in Chapter 2.3, it may be very difficult to create a software recognizing incoherent issues from data. Nowadays, it has become accessible to train a computer to solve similar problems. In 1959, Arthur Samuel defined machine learning as a field of study giving computers the ability to learn without being explicitly programmed. In other words, machines are not specifically instructed in advance to perform each case in a particular way, but they independently learn from data. Algorithms that explore the data to gradually learn, are utilized to improve performance in a given task with more and more experience. While the model develops and improves the way of describing the available data, it can better predict the end results (Samuel, 1959). The field of machine learning is formed by mathematical models and algorithms required to efficiently classify the data (Dietterich, 1997; Bishop, 2006; Hastie, Tibshirani, and Friedman, 2009).

In most cases, including this study, machine learning refers to methods based on supervised learning. Other machine learning methods include unsupervised learning mainly investigated by data mining and knowledge discovery (Ghahramani, 2004) as well as reinforcement learning which learns by an agent from direct interaction with its environment without being advised of correct answers in advance (Sutton and Barto, 1998). In supervised learning, a training dataset is employed from the initial data, and used to identify patterns by matching or resembling annotated regularities in the data. Thus, an inferred function is produced by analyzing training samples including both features and labels. In other words, labels are assigned to each data point and therefore, initial data points and the related known labels are required to teach a supervised learning model (Goodfellow *et al.*, 2016).

While using most of the data for training the model, the rest is used for testing. A test dataset is applied to indicate the capability of the model in predicting a certain output. Therefore, the interest is the performance of the algorithm on unseen data representing the real world. Thus, the result describes the ability of the algorithm in managing the future tasks. Figure 13 illustrates this grouping. The whole classified initial dataset is split into two smaller sets, namely training and testing datasets. Both sets have different, but important purposes in developing the machine learning model (Kriesel, 2007; Goodfellow *et al.*, 2016). As stated in Chapter 1.2, only supervised learning, specifically corrective learning or error correction, is relevant to this study. Hence, the other subfields of machine learning are not covered in more detail.

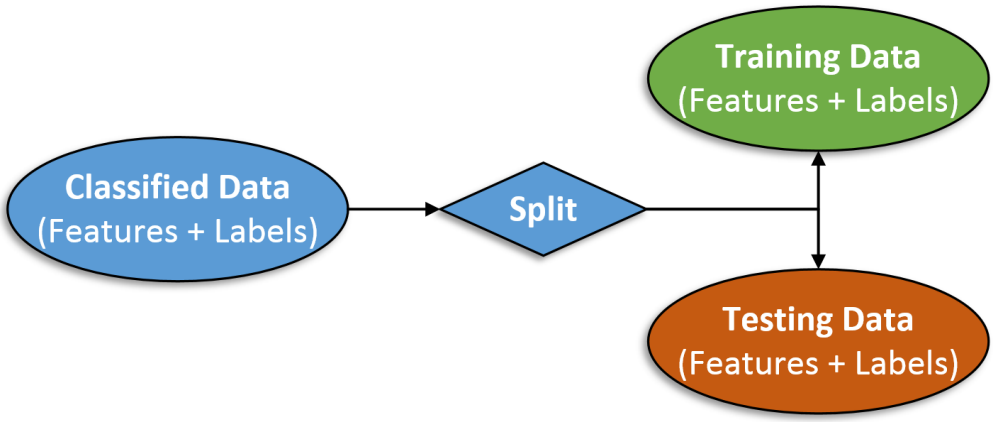


Figure 13 An initial dataset is divided into two different sets in supervised learning (Adapted by the author from Goodfellow *et al.*, 2016)

Simple machine learning algorithms, such as logistic regression and naïve Bayes, can recommend caesarean delivery if suitable (Mor-Yosef *et al.*, 1990) as well as separate legitimate e-mails from spam e-mails (Sun, 2009), respectively. Their performance depends heavily on the representation of the provided data, whereas in hardest tasks, there are factors of variation influencing every feature of data. Because of this challenge in representation learning, a method called *deep learning* has been developed allowing "the computer to build complex concepts out of simpler concepts" (Goodfellow *et al.*, 2016, p. 5).

The main difference of deep learning, belonging to the most advanced AI discipline (i.e., representation learning), compared to other disciplines are briefly discussed as follows. In rule-based systems, there are only hand-designed programs, whereas classic machine learning includes mapping from features based on the manually-designed features. There are two learning methods differing from each other only by the number of layers in the representation learning. Deep learning includes both simple features and additional layers of more abstract features following by mapping. Mapping from features implies the ability to learn from data (Goodfellow *et al.*, 2016).

The description of the ways the machine learning system should process an example is considered a machine learning task. An example, typically represented as a vector $\boldsymbol{x} \in \mathbb{R}^n$, is a collection of features quantitatively measured from an object or event that is wished to be processed by the machine learning system. Concerning this study, in a machine learning task called classification the objective is to specify a correct category to which an input belongs. In case a feature can concurrently have two or more categories, the task is called a multi-label classification (Goodfellow *et al.*, 2016). Especially in a multi-label classification, there are several measures, each of them providing slightly different information about the accuracy. Depending on the desired behavior of the system, a convenient measure can be chosen to represent the most appropriate information, even though the choice of performance measure is recognized to be difficult (Kafrawy, Mausad, and Esmail, 2015).

An important field of machine learning, also involved in this research, is pattern recognition. Pattern recognition is used e.g. for the identification of faces, objects, words and melodies. It aims to model patterns and regularities found in data, while machine learning focuses on maximizing the rate of recognition. Given that they are related to each other, the pattern recognition occurs implicitly when the weights of neural networks are modified (Kocaleva *et al.*, 2016).

A performance measure must be designed to evaluate algorithm's abilities, which in the case of a classification task might imply the accuracy. There are many ways to measure accuracy of a model. Precision refers to the portion of the selected items being relevant, while recall results in the portion of the relevant items which have been selected. Accuracy evaluates "the percentage of correctly predicted labels among all predicted and true labels" (Kafrawy *et al.*, 2015, p. 3). In multi-label cases, which this study also represents, Hamming loss function is widely used to measure the ability of the algorithm (Kafrawy *et al.*, 2015).

A confusion matrix is designed to illustrate different possibilities for answers in binary classification tasks. When there are only two possible answers representing correct and incorrect predictions, the confusion matrix shown in Figure 14 emphasizes the corresponding results. In machine learning, actual class is usually called "ground truth". Multi-label classifications are more complicated since there may be various classes

associated with a feature. Thus, predicted classes may also be partly correct. However, the confusion matrix is a useful method to illustrate the error types a classifier can also generate in a multi-class case, specifically, false positive and false negative errors. False positive means the classifier has predicted a label even it is not true, whereas false negative implies that a class is not predicted while it should be (Weizhong and Goebel, 2004).

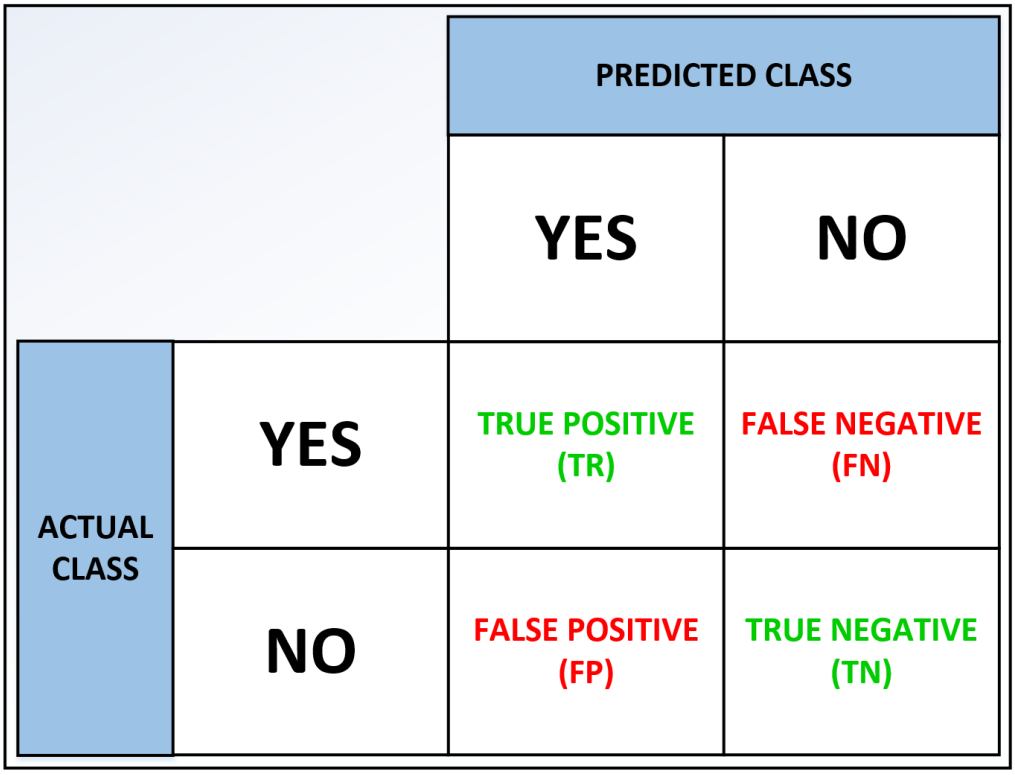


Figure 14 Confusion matrix for 2-class classification problems (Adapted by the author from Weizhong and Goebel, 2004)

According to Kafrawy *et al.* (2015, p. 3), "Hamming loss evaluates how many times an example-label pair is misclassified, i.e., label not belonging to the example is predicted or a label belonging to the example is not predicted". Smaller hamming loss value corresponds to the better performance. Hamming loss (HL) is defined below in Equation 1. In the equation, Δ stands for the symmetric difference between two sets, N is the number of examples and Q is the total number of possible class labels, as specified in the article (Kafrawy *et al.*, 2015).

$$HL(h) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Q}|h(xi)\Delta yi| \quad (1)$$

In practice, if the algorithm only predicts one label compared to two labels in the ground truth, the accuracy is one correct out of two. Table 7 demonstrates this logic. As Hamming loss is the fraction of labels that are incorrectly predicted among all labels, and always between 0 and 1, in the multi-label classification the accuracy can be considered as $1 - Hamming\ loss$. Hamming loss is a loss function having the optimal value zero. In that situation, every predicted label would be correct when compared to the ground truth set by a human.

Table 7 Example of the accuracy calculation

Predicted Label	Ground Truth	Score
Label 1, Label 2	Label 1, Label 2, Label 3	66 %
Label 1	Label 1, Label 2, Label 3	33 %

Concerning the confusion matrix and measuring the ability of a binary classifier, a receiver operating characteristic (ROC) curve can be plotted. It illustrates the ability of the classifier to avoid errors which it may perform, specifically false positive error and false negative error, as shown in Figure 14. False positive means the classifier has predicted a label even if it should not have. Conversely, false negative represents an opposite situation (Zweig and Campbell, 1993). Thereafter, the error rates, namely false negative rate (FNR) and false positive rate (FPR) can be calculated. ROC curves are plotted by setting true positive rate (TPR) on the y-axis and false negative rate on the x-axis as illustrated in Figure 15. The ideal classifier would have as high true positive rate as possible and low false positive rate. The curve C represents a well-performing classifier as it is located in the upper-left corner; thus, the true positive rate overcomes the false positive rate (Weizhong and Goebel, 2004).

In addition to the curves in a binary classification, the area under the curve (AUC) can be determined. That is considered an effective and combined measure of classifier's performance (Bradley, 1997). It measures both sensitivity and specificity. When the AUC equals to one, it implies that both errors (false positive and false negative) are zero. That is a theoretical case since it is very unlikely to happen in practice. However, it is stated that the AUC closer to one indicates better performance of the test. While the diagonal line represents the case in which both rates are identical, that is, the proportion of correctly classified samples equals to the proportion of incorrectly classified samples, the ROC curve afar from the diagonal line indicates better performance of the classifier (Kumar and Indrayan, 2011).

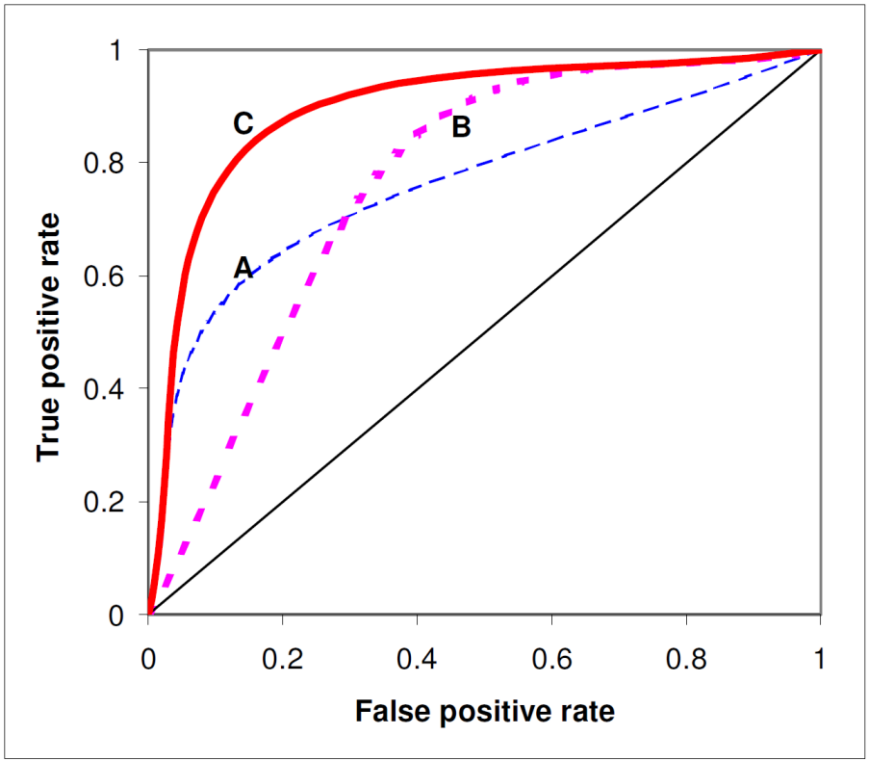


Figure 15 Typical ROC curves (Weizhong and Goebel, 2004)

2.3.2 Artificial Neural Networks

To manage information without being explicitly programmed, McCulloch and Pitts (1943) presented a concept of artificial neural network (ANN). In ANN, a set of nodes (computational units) and connections between them are utilized while seeing the nodes as "artificial neurons". The nodes, also called as perceptrons, receive inputs and process them to obtain an output by, for instance, summing the inputs or utilizing another network inside a node. Artificial neural networks (hereinafter referred to as neural networks) are inspired by the brain, and utilize algebra, functions, vectors as well as differential calculus, though mainly involving basic matrix operations and nonlinear regression. Only important basics relevant to the study are described because the neural network is an extremely broad concept (Rumelhart and McClelland, 1986; Rojas, 1996; Das, 2016).

The connection strengths and the ways of transforming input into output determine the behavior of an artificial neuron. Transformation is performed through an activation function located in every neuron multiplying each input with the associated parameter. The connections determine the information flow between the nodes. The flow can either be unidirectional (Bar-Yam, 1997) or bidirectional (O'Reilly *et al.*, 2012). In the neural network, learning is due to the modification of synaptic weights that is parallel with a human way of learning from experience (O'Reilly *et al.*, 2012; van Gerven and Bohte, 2018). The parameters are automatically adjusted in a learning algorithm finding the best combination for the solution of a given problem (Rojas, 1996).

An abstract neuron and its structure including an extra weight are illustrated in Figure 16. Each connection transmits a real value x_i, which is multiplied by an associated weight w_i. Generally, "**w**" represents a vector of weights or parameters which control the behavior of the system, involving both positive and negative numbers. Thereafter, the products are integrated and a primitive function, also known as an activation or a squashing function, is evaluated. An arbitrarily selective activation function utilized in each neuron aims to transform its input into a precisely defined output. The extra weight $-\theta$ connected to the constant 1 in Figure 16 is called a bias of the element. It is considered as a threshold of the neuron, and its import will be discussed later with the activation of the neuron (Rojas, 1996). As stated by Rojas (1996, p. 23), "artificial neural networks differ mainly in the assumptions about the primitive functions used, the interconnection pattern, and the timing of the transmission of information".

The network in Figure 17 can be considered as a function Φ, also called a network function, of which components are the primitive functions considered neurons. The network functions depend on the weights according to which they change. The following three elements are stated to be important in artificial neural networks: "the structure of the nodes, the topology of the network and the learning algorithm used to find the weights of the network" (Rojas, 1996, p. 24).

The parameters are pursued to be adjusted "in an optimal manner to reflect the information known and to extrapolate to new input patterns" which will be entered. Adjusting the network's parameters, that is, weights and biases, is called learning. Unfortunately, the network itself is a black box in many cases. Therefore, a self-organizing process produces a desired output from a certain input, thus, an n-dimensional real input is mapped to an m-dimensional real output (Rojas, 1996).

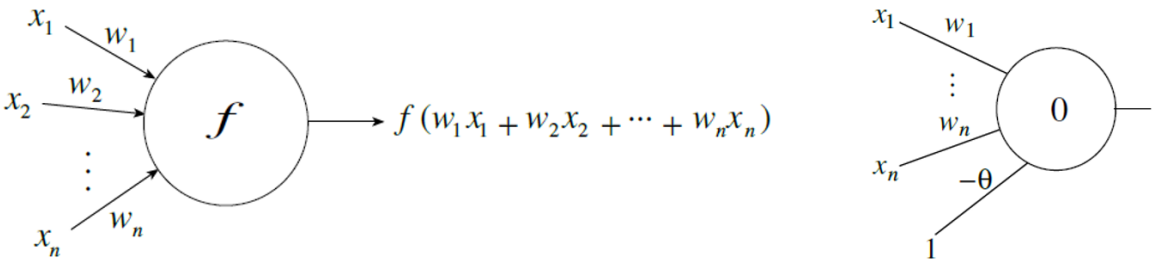


Figure 16 An abstract neuron and a perceptron with a bias, respectively (Rojas, 1996)

Already in 1961, Frank Rosenblatt (1961, p. 291) introduced an idea of employing "a procedure which randomly varies the value of every connection, independently of the others" until there is no error. Deep learning is often utilized based on the feedforward neural network (FNN), also known as a multilayer perceptron (MLP). The MLP is a multilayer network, a mathematical function composed of many simpler functions, mapping input values to output values – that is, the objective is to approximate a function f^*. For instance, for a classifier, $y = f^*(\boldsymbol{x})$ maps an input vector "**x**" to a category "y". In feedforward network, the mapping is defined as $y = f(\boldsymbol{x}; \boldsymbol{w})$, in which the value of the parameters $\boldsymbol{w}$ are trained to result the best function approximation (Hinton, Rumelhart, and Williams, 1986; Goodfellow *et al.*, 2016).

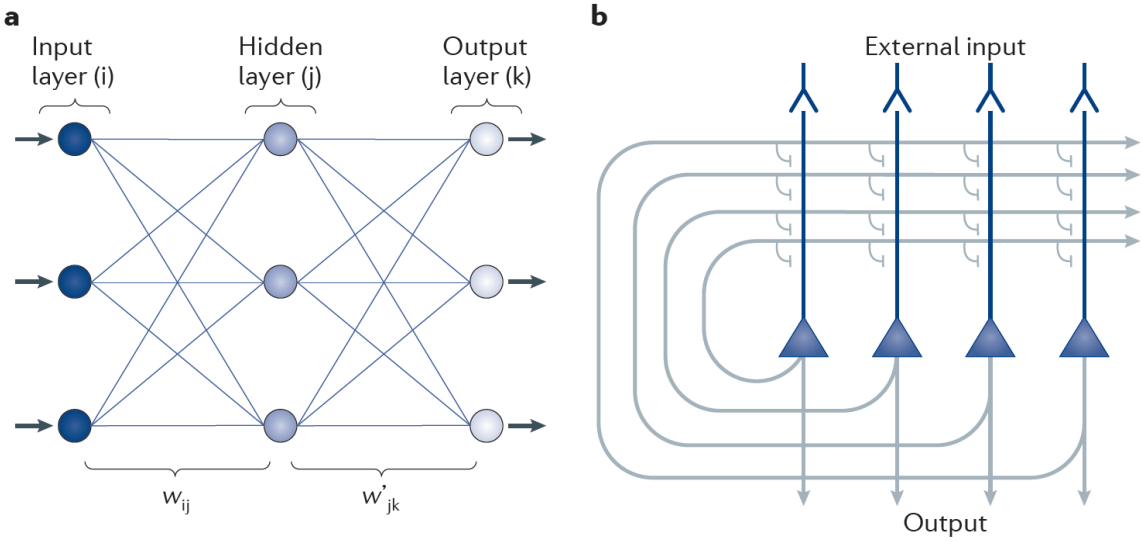


Figure 17 Artificial neural networks: (a) feedforward neural network and (b) recurrent neural network (Yuste, 2015)

In feedforward neural networks (also called as feedforward networks), information flows only in one direction as illustrated in Figure 17 (a). A weight is assigned to each connection between neurons in different layers. The network function is directly evaluated with a network of primitive functions. The feedforward network shown in Figure 17 illustrates a multilayer perceptron with three sequential layers of neurons, in which each neuron of previous layer is connected to every neuron of the next layer. However, the feedforward network cannot record previous results to improve computations by reusing the calculated outcomes. The cycles shown in Figure 17 (b) store and reuse signals. Including feedback connections results in a formation of recurrent neural network (RNN) which is capable of keeping track of previous results and storing bits to be reused (Rojas, 1996; Yuste, 2015).

The primitive function is utilized to determine whether a neuron becomes active or inactive due to the result of the node. The value of the activation function is considered activation which should be ranged 0 to 1, simulating the way the biological neuron works in estimating the positiviness of the relevant weighted sum. In the case of the multi-label classification, logical values 1 (true) or 0 (false) are not applicable, but rather a continuous distribution. The bias of each perceptron determines the meaningful activation of the perceptron, because the bias has an influence on the weighted sum (Rojas, 1996; Yuste, 2015). Currently, two common non-linear activation functions include s*igmoid* and *rectifier*. When the latter one is employed in a unit, it is called Rectified Linear Unit (ReLU). Activation functions are essential when considering the amount a certain neuron may affect to the result. The relevant activation functions are shortly described as follows (Jarrett *et al.*, 2009; Glorot, Bordes, and Bengio, 2011).

The logistic sigmoid function in Equation 2 has a characteristic "S"-shaped curve, also called sigmoid curve. As presented in Figure 18, this function squashes the output into the interval [0, 1] which is considered a probability. One major benefit of the sigmoid function is that in addition to a smooth gradient, very negative inputs result in close to zero and very positive inputs close to one. There are various combinations of the sigmoid function available but Equation 2 represents the basic one (Goodfellow *et al.*, 2016).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

The rectified linear unit is defined by the activation function. When applied to the output of a linear transformation, the result is a nonlinear transformation. Even though the function consists of two linear pieces, it is a nonlinear function including beneficial properties of both linear and nonlinear functions. One of these aspects is that gradient-based methods can be easily applied to optimize models. Because the ReLU is simpler than the sigmoid function, it has been recognized to perform better in deep neural networks (Goodfellow *et al.*, 2016).

$$g(z) = max\{0, z\} \tag{3}$$

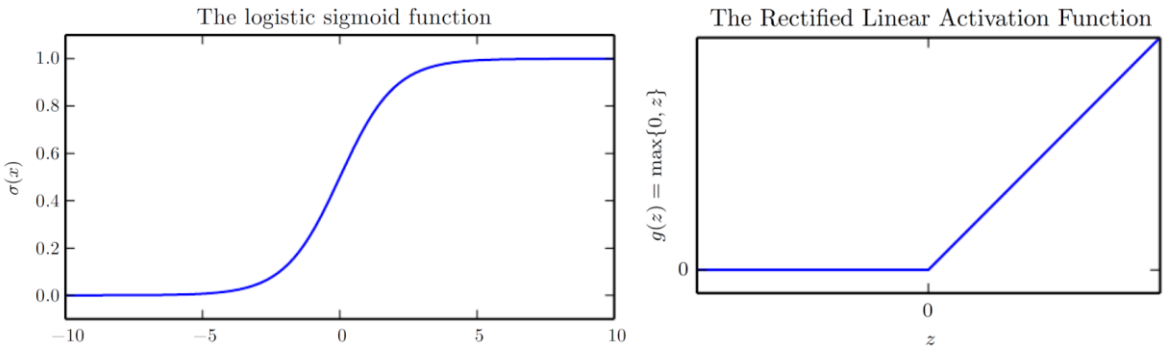


Figure 18 The logistic sigmoid function and the rectified linear activation function, respectively (Goodfellow *et al.*, 2016).

An objective function essentially influences building a machine learning algorithm during the selection of a dataset, optimization procedure and modelling. The object function, also called criterion, is either minimized or maximized. In the case of minimization, the function may be called the cost, loss or error function. Generally, there is at least one term in the function causing the learning process to perform statistical estimation. Goodfellow *et al.* (2016) state that the most common cost function is the negative log-likelihood, of which minimization causes maximum likelihood estimation. In nonlinear model, the function is optimized in open form requiring an iterative numerical optimization procedure, such as gradient descent (Rojas, 1996; Goodfellow *et al.*, 2016). The optimization is very challenging, especially in the case of the multidimensional input. Finding a global minimum (Cauchy, 1847) among various possible local minima and saddle points is facilitated by accepting a very low value relatively close to the global minimum. Furthermore, partial derivatives are utilized to individually measure the changes of multiple inputs (Goodfellow *et al.*, 2016).

As mentioned at the beginning of Chapter 2.3.1, the performance of the model is tested against a separate testing dataset not used for training, specifically, a design matrix of *m* example inputs with "a vector of regression targets providing the correct value of y for each of these examples". If the vector of regression targets are defined as $\boldsymbol{y}^{(test)}$ and the predictions of the model as $\hat{\boldsymbol{y}}^{(test)}$, the performance of the model can be measured by computing the mean squared error of the model on the test set as defined in Equation 4 (Goodfellow *et al.*, 2016, p. 108).

$$MSE_{test} = \frac{1}{m}\sum_{i}\left(\hat{\boldsymbol{y}}^{(test)} - \boldsymbol{y}^{(test)}\right)_i^2 \tag{4}$$

To train a machine learning algorithm, the design matrix of inputs $\boldsymbol{X}^{(train)}$ with the targets $\boldsymbol{y}^{(train)}$ refer to the training data, that is, features and the associated labels. The algorithm should improve the weights $\boldsymbol{w}$ reducing the mean squared error of the test set while observing the training set. The mean squared error on the training set may be minimized by setting the gradient to equal zero, and noticing that the error decreases as the Euclidean distance between the predictions and the targets decreases. Thus, no error exists when $\hat{\boldsymbol{y}}^{(test)} = \boldsymbol{y}^{(test)}$ (Goodfellow *et al.*, 2016).

$$\frac{1}{m}\nabla_w\left\|\boldsymbol{X}^{(train)}\boldsymbol{w} - \boldsymbol{y}^{(train)}\right\|_2^2 = 0 \tag{5}$$

The cost function estimates the performance of the model, involving an average cost of all training data. It is approximated by comparing the estimated predictions against the ground truth, that is, the known values of **y** (Bishop, 2006). The algorithm learning its weights and biases using the gradient descent method in the cost function is known as backpropagation algorithm. In other words, neurons are organized in layers sending signals forward and propagating the errors backwards (Rumelhart and McClelland, 1986; Rojas, 1996). The cost function combines each weight and bias, and results in a single number called the cost. The gradient of the cost function notifies the extent of the change of the weights and biases, and the direction to cause the fastest change (Rojas, 1996; Goodfellow *et al.*, 2016). According to Nielsen (2015), the biggest neural networks have cost functions depending on billions of weights and biases combined in a complicated way.

Two key challenges in machine learning include underfitting and overfitting which occur when the model cannot "obtain a sufficiently low error value on the training set" and "when the gap between the training error and test error is too large", respectively (Goodfellow *et al.*, 2016, p. 111). Studies indicate that especially in deep neural networks (DNNs), the quality of the training data has an evident impact on the accuracy of the algorithm. Consequently, when the model is trained with unstructured (randomly labeled) data, the cost function struggles to find the local minima compared to training on the structured (properly labeled) data. The local minima should be easier determined in the case of the structured data (Choromanska *et al.*, 2015; Arpit *et al.*, 2017). The convergence of the backpropagation algorithm requires certain conditions on the network architecture and the learning environment (Gori and Tesi, 1992).

2.3.3 Natural Language Processing

A key subfield of machine learning in requirements classification tasks is natural language processing to manage naturally written text. According to Sebastiani (2002, pp. 1–2), the automated categorization (or classification) has been an interest of information retrieval since 1960s. However, it only became popular in the early 1990s as content-based document management tasks gained a notable position because of the increased digital documents available and the need of flexibly accessing them. Text categorization (also text classification or topic spotting) belongs to these tasks, being "the activity of labeling natural language texts with thematic categories from a predefined set" (Sebastiani, 2002, p. 1). Utilization of machine learning improved these tasks since an automatic text classifier could be built by training the characteristics of the categories of interest from a set of pre-classified documents (Sebastiani, 2002).

The idea of natural language processing is to program computers to process sentences from either spoken or written languages, analyzing the morphology, lexicography and even the semantics of whole sentences. Also referred to as computational linguistics, natural language processing aims to teach the computer the way of speaking and understanding sentences by defining and describing patterns from natural language. The advantages of the approach include a better accuracy compared to a human expert, and savings, since the classifier can be constructed based on the regularities in the data. Interesting investigations include the way people are able to comprehend the meaning of a spoken or written sentence, and the manner of understanding incidents, time, places, assumptions, beliefs or facts (Sebastiani, 2002; Kocaleva *et al.*, 2016).

Formally, the objective of the text categorization is to approximate an unknown target function $\breve{\Phi} : \mathrm{D} \times \mathrm{C} \rightarrow \{\mathrm{T}, \mathrm{F}\}$ describing the classification ways of documents by means of a function $\Phi : \mathrm{D} \times \mathrm{C} \rightarrow \{\mathrm{T}, \mathrm{F}\}$ called a classifier, such that the difference between the functions is minimized. In the functions, D and C are a domain of documents and a set of predefined categories, respectively. Depending on the amount of categories to be assigned to each $d_j \in D$, the classification may be called a single-label or multi-label case (Sebastiani, 2002).

Commonly, ”NLP applications are based on language models that define a probability distribution over sequences of words, characters or bytes in a natural language”, as stated by Goodfellow *et al.* (2016, p. 463). As a result, the goal is to learn “the joint probability function of sequences of words in a language” (Bengio *et al.*, 2003, p. 1137). In other words, the language model is a way of understanding the next word given the context of previous ones. The importance of a domain-specific strategy is highlighted in order to achieve excellent performance and to properly scale to large applications. Natural language is typically regarded as a sequence of words that results in an extremely high-dimensional and sparse discrete space. Out of several strategies developed for making models of such a space efficient, *Long Short-Term Memory* (LSTM) important with respect to the study, is next briefly discussed (Goodfellow *et al.*, 2016).

Modelling long sequences induced major challenges until an efficient gradient-based method called LSTM was introduced by Hochreiter and Schmidhuber (1997) to advance sequence modeling tasks, including natural language processing tasks. In comparison with other similar applications, LSTM learns faster and is the first model solving complex, artificial long time lag tasks never solved before (Hochreiter and Schmidhuber, 1997). LSTM utilizes a recurrent neural network architecture in conjunction with an appropriate gradient-based learning algorithm. A LSTM unit consists of several other neural networks, having individual tasks, such as ignoring, forgetting and selecting words. The idea of LSTM is being capable of remembering the events occurred since many time steps. It can look back and decide whether to forget or add certain words into the prediction based on the history of used words in sentences. For this reason, a copy of predictions is stored for the next prediction. Separate neural networks learn to forget and select correct words based on the previous predictions as well as immediately irrelevant words being ignored to avoid interrupting the prediction and selection. While each activity has its own neural network, they also have a specific logistic squashing function and gating activity associated with the activity (Hochreiter and Schmidhuber, 1997).

The architecture of RNN is applied in the LSTM models as a short-term memory of RNN is combined with a long-term memory of LSTM presented in Figure 19. The memory cell is controlled by gate networks (here referred to as delta (δ) and sometimes sigma (σ)), specifically a forget gate network (*f*), an input gate network (*i)* and an output gate network (*o*). They control the amount of passing information, scale the input block *u* to the internal cell and aim to control the output of the internal cell, respectively (Xia *et al.*, 2018). In the LSTM network, a hidden layer consists of nodes considered as memory block cell assemblies (Monner and Reggia, 2012). In the LSTM cell, the memory generation is based on “the input word x_t and the past hidden state h_{t-1} to generate a new memory c_t which includes aspects of the new word” (Mohammadi *et al.*, 2019, p. 14). As it can be seen in Figure 19, the hidden state h_t moves vertically up and horizontally from left to right. All the operations within one cell generate the vector which includes essential information of the context (Mohammadi *et al.*, 2019).

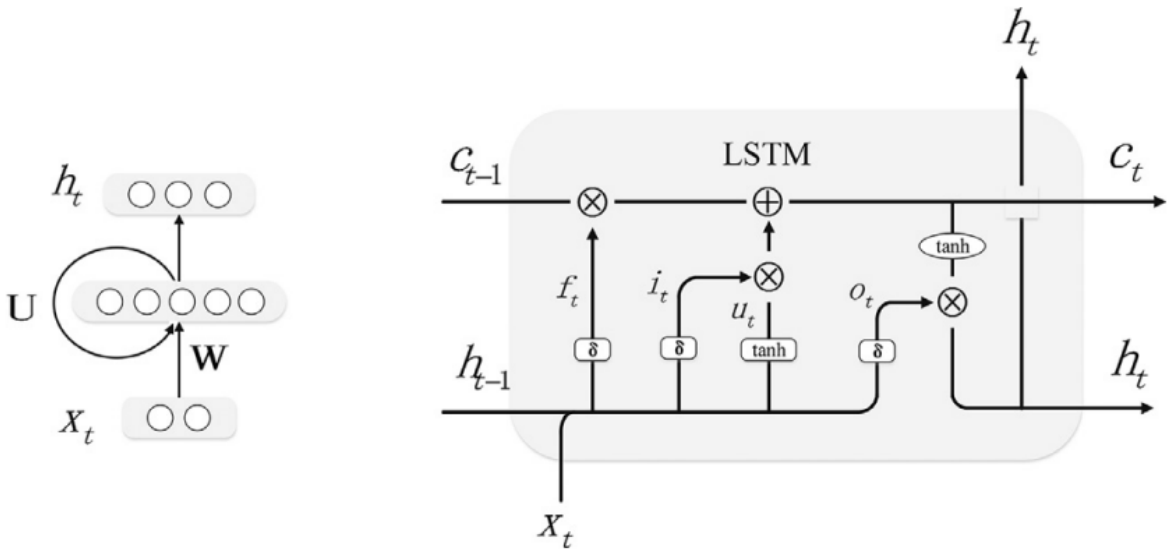


Figure 19 Recurrent neural network (left) is utilized in the LSTM (right) (Xia *et al.*, 2018)

Based on a common text classification process (Ikonomakis, Kotsiantis, and Tampakas, 2005), a tailored process flow applicable in the case study is represented in Figure 20. At the beginning of the process, a document is read, text is segmented into linguistic units, such as words, punctuation and numbers. While documents are presented by an array of words, there are words called stop words, such as auxiliary verbs, conjunctions and articles, which were formerly removed as a part of the process (Ikonomakis *et al.*, 2005). Currently, they are retained since stemming (Brank *et al.*, 2002; Han *et al.*, 2004) and removing stop words might destroy important words. Feature transformation maps a set of values for the feature to a new set of values to make the representation of the data more suitable (Ikonomakis *et al.*, 2005). Text categorization methods, such as support vector machines (SVM) and k-Nearest Neighbor (k-NN) approaches based on Latent Semantic Indexing have been discovered performing better than the other methods (Leopold and Kindermann, 2002; Cardoso-Cachopo and Oliveira, 2003).

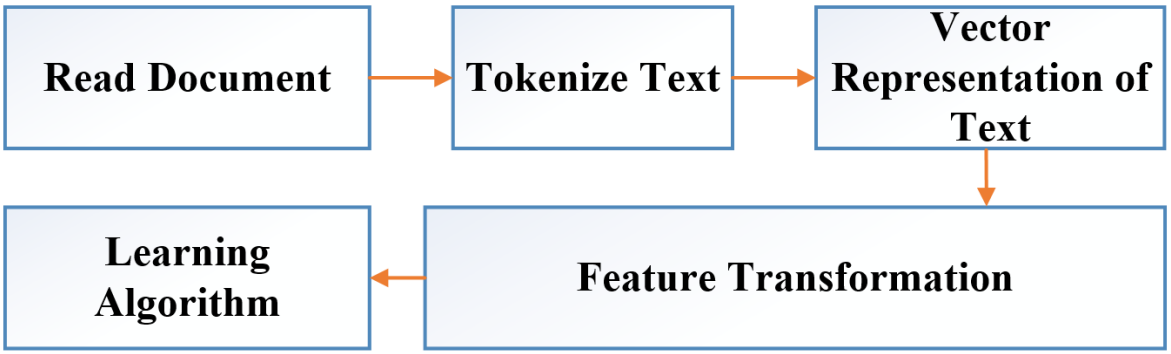


Figure 20 Text classification process applicable in this study (Adapted by the author from Ikonomakis *et al.*, 2005)

In text classification, each word is associated with a distributed word feature vector consisting of real values. Bengio *et al.* (2003, p. 1139) state that "the feature vector represents different aspects of the word: each word is associated with a point in a vector space". When every word of the sentence has a feature vector, the joint probability function of word sequences is expressed. Therefore, the word feature vectors and the parameters of the probability function are learned. Bengio *et al.* (2003) continues that "the probability function is expressed as a product of conditional probabilities of the next word given the previous ones". The multi-layer neural network can be utilized to provide the prediction of the next word when the previous words are known in a sentence. In case of knowing that two words are semantically and syntactically close to each other, they are expected to have a similar feature vector. Consequently, due to a smooth probability function, a small change in the features will induce a small change in the probability (Bengio *et al.*, 2003, pp. 1139–1140).

To improve a deep learning task, transfer learning can be utilized by using knowledge gained while solving one problem and applying that to a different but related problem. For instance, if a network has learned to recognize cats, the knowledge of that network could be applied to classify tigers (Tan *et al.*, 2017) or explicate a minor language based on the knowledge achieved from a major language (Zhou *et al.*, 2016). This knowledge transfer may improve the performance of learning without requiring data-labeling efforts, such as adapting a certain classification model trained on some products to help learn the classification models for other products. While traditional machine learning separately solves specific problems, transfer learning utilizes useful information of other task available to solve another and different task. Therefore, the transferred knowledge has to be common between the domains. Different learning methods can be used based on availability of labels in a target or source domain (Pan and Yang, 2010).

2.3.3.1 Practical Applications of NLP

AI techniques have been observed to be an advantageous solution to automate certain processes of the requirements phase. Thus, the human intervention may be minimized in specific requirement tasks, such as elicitation, analysis and modeling (Sharma and Pandey, 2013; Tripathy, Agrawal, and Rath, 2014). This chapter briefly highlights examples of the significant contributions in the related area.

An approach for the automatic classification has been presented being capable of classifying requirements into two categories, namely "requirement" and "information". The approach based on convolutional neural networks (CNNs) and the capability to analyze natural language, could be used both for classifying unseen content or verifying the categorized documents to identify potential incorrect classifications. The CNN is a subclass of the neural networks having a specific network architecture (le Cun, 1989). The preliminary results of the binary classification task are promising and reported to be comparable to other tasks also employing CNNs (Winkler and Vogelsang, 2016).

Natural language processing is recognized to advance requirements specification analysis, such as requirements checking. The same requirements for the ontology have been determined as mentioned in the course of this research: correctness, consistency, completeness and unambiguity (Bures *et al.*, 2012). The integration of software engineering and natural language processing has been widely investigated (Yalla and Sharma, 2015).

Automatic quality assessment methods have been developed for textual requirements in the software requirements specification documents. While one classifier aims to detect ambiguous and unambiguous texts at the surface-level of understanding (Ormandjieva, Hussain, and Kosseim, 2007), the other one classifies the manually formed requirements. Thus, the possibility of the requirements being precisely written can be noticed based on the categorization (Tamai and Anzai, 2018). NLP can also be used to improve the information retrieval process, specifically automatic search in requirements engineering. Instead of a key-word based search, the search engine could be aware of semantic relations (Di Martino and Cretella, 2013).

Models called Zero-shot Learners have been studied being able to predict unseen classes. They are designed for text categorization as a binary classification problem. The model predicts a possible relation of a given sentence to a certain group, in contrast to a classifier categorizing the sentence into one or multiple classes. Three different architectures, two of which included long short-term memory networks (LSTMs) were tested in the study. The approach has been recognized to progress the development towards general intelligence in natural language processing. However, the reported accuracies have not yet achieved the level of the supervised models (Pushpankar and Muktabh, 2017).

In sentiment analysis, a fine-grained task called aspect-level sentiment classification has been recognized to explore the connection between an aspect and the content of a sentence. Therefore, an Attention-based Long Short-Term Memory Network for aspect-level sentiment classification has been proposed, being capable of capturing important information in response to a given aspect. Contrary to the standard LSTM, the model is able to detect the important parts of the sentence while concerning various aspects (Wang *et al.*, 2016).

The ability of AI to use natural language processing to analyze regulation data is believed to provide extensive benefits. Extensive investments are reported to be performed every year, such as addressing compliance. Therefore, new applications of AI are investigated to facilitate this challenge in regulatory analysis. In addition, converting unstructured data of legal agreements into structured data is reported to be possible more quickly and accurately than a human expert can perform (Broadridge, 2017).

3 Data and Methods

The aim of this chapter is to create an understanding regarding methodological choices and the execution of this research. The execution of the study involves research philosophy presented in the form of research onion formed by Saunders, Lewis, and Thornhill (2007), according to which the reflected assumptions have been adopted. The applicable parts of the union are illustrated in Figure 21. The adoption of the research onion aims at creating a scientific basis of scrutiny by increasing the perspective when moving deeper in the onion. This chapter describes the way the study has been performed in practice, and the new and essential data collected.

This study is based on the deductive approach as suitable theory was first familiarized with and the research questions were set, following by data collection, findings, discussion, and conclusion. As a strategy, case study design is utilized to involve extensive study of special cases. Mixed methods were chosen to be used because of the nature of the research questions and objectives. This means that the capabilities of the algorithm are analyzed by using both the quantitative and qualitative research methods. The time horizon is a cross-sectional design since this case study is only planned to be performed once, and possible further investigations should improve the knowledge and tools. Finally, data collection and data analysis are carried out.

An AI principle represented by Jung (2018, p. 4) states the following: "Based on the perceived environment, compute actions (decisions) which allow to maximize a long-term return". Therefore, there is a need for the precise definitions of "perceived environment", "actions" and "return" required to adapt the principle. In terms of the study, perceptions are given by requirements, actions amount to the analyses and classifications of the requirements, whereas, return is measured in reduction of working hours typically used in individually processing the specifications.

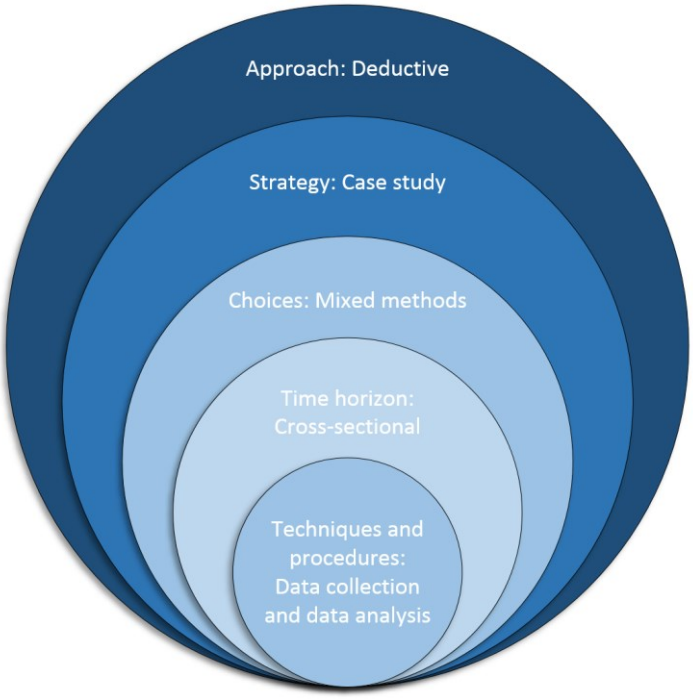


Figure 21 The research onion for this study

3.1 Data Collection

An initial data collection of the YVL Guides (STUK, 2018a) was selected to be analyzed in this study. The PDF files of the guides are available in Finnish, Swedish and (British) English on the STUK website. However, only the English versions were used and exported from the Fortum tabular database into Excel files. The data collection hierarchy is represented in Figure 22 in which the content equality is emphasized between the Excel and PDF files, that is, the requirements are the same and in the same order despite the format itself. The Excel format was chosen because it enabled the easy insertions of attributes for each requirement.

The initial collection of the classified data comprises 10 YVL Guides of which degree of utilization varies from approximately half to complete set of requirements by reason of the efficient data collection, that is, to avoid labeling extra attributes. Therefore, not every guide was completely analyzed and classified, but only a reasonable portion to have a suitable amount of each label. The utilized portion of each guide is listed in Table 8. As an observation, the abbreviation of *I&C* in the title of YVL Guide E.7 refers to instrumentation and control. The numbers in the table have been rounded to the nearest whole number. Due to the machine learning task, collecting the learning data requires almost the same amount of labels for each category. It is especially important to have an equal amount of requirements represented to a greater or lesser extent so that the algorithm can evenly learn to predict each class. As mentioned in Chapter 2.3.2, there is a common care that the algorithm would learn to predict certain classes overly well (overfitting) or poorly (underfitting).

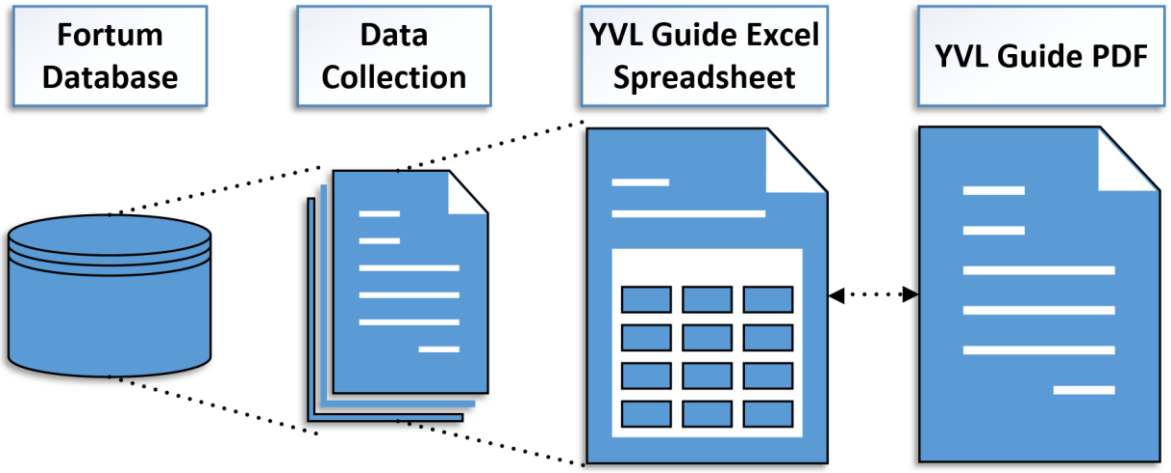


Figure 22 Data collection hierarchy from the database to a spreadsheet

Table 8 YVL Guides used in the initial requirements categorization

YVL Guide	Title	Degree of Utilization
YVL B.1	Safety Design of a Nuclear Power Plant	100 %
YVL B.4	Nuclear Fuel and Reactor	100 %
YVL B.5	Reactor Coolant Circuit of a Nuclear Power Plant	100 %
YVL E.3	Pressure Vessels and Piping of a Nuclear Facility	55 %
YVL E.4	Strength Analyses of Nuclear Power Plant Pressure Equipment	50 %
YVL E.6	Buildings and Structures of a Nuclear Facility	57 %
YVL E.7	Electrical and I&C Equipment of a Nuclear Facility	100 %
YVL E.8	Valves of a Nuclear Facility	73 %
YVL E.9	Pumps of a Nuclear Facility	45 %
YVL E.10	Emergency Power Supplies of a Nuclear Facility	42 %

These 10 YVL Guides out of total 47 guides (STUK, 2018a) were sufficient to reach the expected amount of labels for each class. In addition, one guide was used in both training the language model and testing the classifier. Specifically, the listed 10 YVL Guides were used for training as well as preliminary testing, and YVL Guide B.2 was used both for the separate test and training the specific language model. The sufficient amount of requirements for each class was originally determined to be 500 requirements in terms of high-level categories, and 100 requirements in terms of subcategories. This was resulted in due to the studies indicating that an algorithm utilizing deep learning methods and less than 10 classes would significantly well start to learn only after 400-500 samples per class (Kessler, Numberg, and Hinrich, 1997; Hancock *et al.*, 2018). However, it should be remembered that the more labels for each class, the more accurate model. Hence, the prediction of the model improves as the amount of training data increases. By considering this effect, the determined objective for the number of samples was a compromise between time available to classify requirements and the probability of achieving suitable results.

Two separate language models were used in the case study. Their arrangements and relations to each other are explained in Chapter 4.3. At this point, only the data used for training each model are presented and listed in Table 9. The pre-trained language model of the algorithm was based on 103 million Wikipedia words which are available online (Merity, 2017). The dataset is called “WikiText-103”. The domain-specific language model, also called as “the Fortum language model”, was achieved by improving the model with the text of the YVL Guides mentioned in Table 9.

Table 9 Data collection for training each language model

Language Model	Data	Degree of Utilization
Wikipedia	28,595 Wikipedia Articles	100 %
Fortum	YVL Guides B.1, B.2 & B.4	100 %

YVL Guide B.4 was only utilized for training the Fortum language model, while YVL Guide B.1 was used for the initial training of the model. Additionally, YVL Guide B.2 was utilized for the first blind test, followed by the second blind test with the UK dataset issued by ONR

(ONR, 2014), as listed in Table 10. The UK Safety Assessment Principles (SAP) dataset is considered similar to YVL Guide B.1 which was completely classified for the initial training and testing phases. Hence, the interest was to examine the ability of the model to recognize similar requirements stated by another authority in a different licensing domain.

Table 10 Data collection for the blind tests

Blind Test	Dataset	Title	Degree of Utilization
1	YVL B.2	Classification of Systems, Structures and Components of a Nuclear Facility	100 %
2	UK SAP	Safety Assessment Principles for Nuclear Facilities	100 %

The hierarchy of the classification is later illustrated in Figure 29, and explained in Chapter 4.1. In the context of this study, a term "requirement" is used to cover all the paragraphs or items of YVL Guides, that is, "heading" and "reference" are also considered the requirements categories. Furthermore, the concept "requirement" equals to each requirement ID or paragraph in the guides, thus, only being the requirements considered as written. This is important to be emphasized because practical experiences demonstrate that the initial requirements (IDs) included in the YVL Guides contain as much as 100,000 requirements when the initial ones have been further analyzed and elaborated.

Unfortunately, the YVL requirements, as many other domestic and overseas demands, are noticeably ambiguous and imprecise. This results in various interpretations when analyzed. Hence, a procedure called the Easy Approach to Requirements Syntax (EARS) has been developed to overcome this challenge (Mavin *et al.*, 2009). Its adaptation relies on each party because the approach is relatively new and developed by a private sector.

The requirements included in the guides are considered native requirements, which means they are adopted as written. The native requirements cover for instance YVL Guides (STUK, 2018a), STUK regulations (STUK, 2018b), Nuclear Energy Act (Ministry of Economic Affairs and Employment, 1987), requirements issued by IAEA (IAEA, 2019a, 2019b) and European Union directives (European Commission, 2018). In contrast, the elaborated requirements imply that they have been elaborated from more general ones having a parent-child relationship (Leffingwell and Widrig, 1999). The analysis of these requirements is the first step once the suitable requirements sources have been elicited in requirements analysis. Only the native requirements from the predetermined and explained sources are employed in this study.

The structure of each Excel file is represented in Figure 23. Due to the simplified test case, titles and headings were considered headings and together with each paragraph, they were analyzed as requirements. Figure 23 illustrates the way each configuration item (CI) ID number and requirement ("text/description") are associated with each other having the necessary attributes. Only three possibilities for high-level categories are shown because the fourth one is "heading" and headings cannot have more than one attribute since they are unambiguous. Furthermore, in case a requirement is provided with a process class, at least one subcategory should be attributed. Chapters 4.1 and 4.2 discuss more about the requirements hierarchy and classifications, respectively. Due to the multi-label case, the maximum number of classes associated with the requirement is limited to eight because

headings appear alone. Each heading informs the context of the paragraphs listed below the heading. Hence, they are particularly important from the classification point of view providing useful information about the subject of the requirements. In addition, each attribute has a separate column as illustrated in Figure 23 facilitating the labeling task due to the ease of labeling only one class into each column.

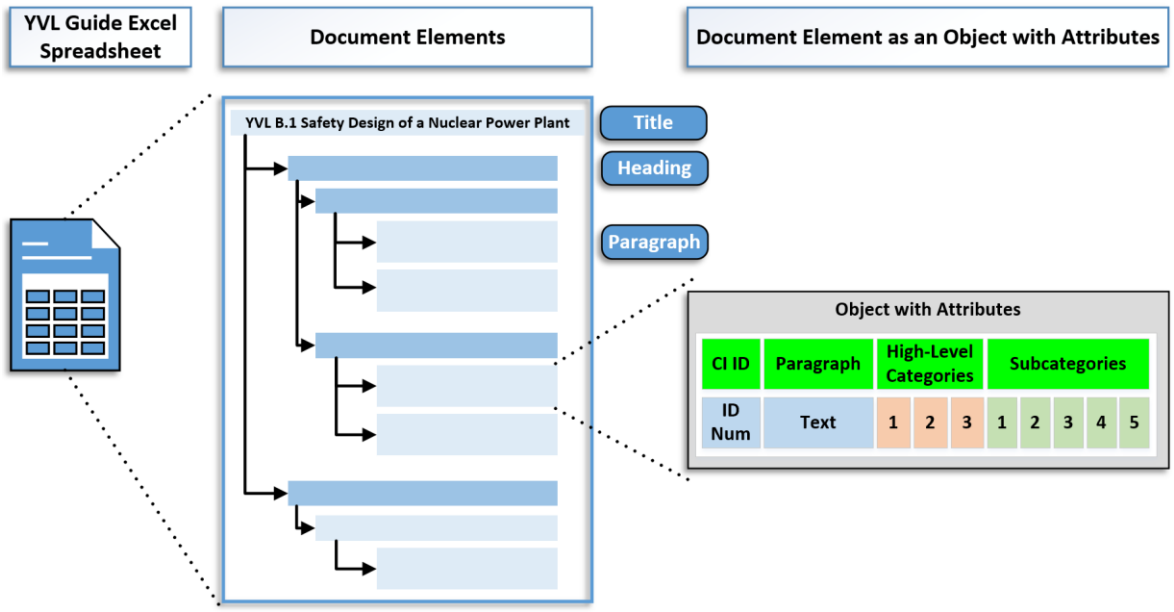
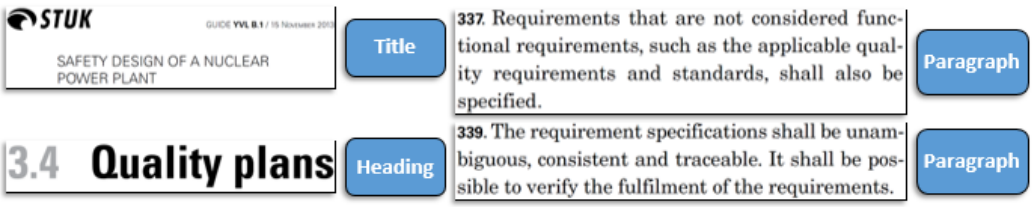


Figure 23 From a whole document through document elements to an individual document element associated with the configuration item ID, paragraph and the relevant categories (Adapted by the author from Karstila, 2013)

Figure 24 emphasizes the way each YVL Guide is considered. The title of the guide and any (sub)chapter is equal to a heading. Each paragraph has a number as well as description (text). Due to the structure and content of the Fortum database, the paragraph number in a PDF file is not the same as configuration item ID in the database. However, the number is originally included in the text of the paragraph as written, and the configuration item ID involves the requirement number as can be seen in Chapter 4.4 Algorithm Classification.

There is a good example of a requirement, namely "332. The quality plan shall present […]", which lists various demands in Figure 24. That illustrates the existence of the requirements consisting of sub-clauses, and still being initially seen as a unity. Each similar demand is considered an entirety and equals to a native requirement. It has already been recognized that these requirements complicate the analysis task despite the analyst, because they may include elements which state various and different demands.

In practice, the substantive attribute names and values relevant to the study are illustrated in Figure 25. That combines both Figure 23 and Figure 24 involving only the relevant items and representing an example of the requirements classification. The requirement is associated with three categories at two different levels, that is, one high-level category and two subcategories. The classification is based on the expert judgement which is further described in Chapter 4.2. However, the presented visualization of the classification is similar to the Excel spreadsheet used in requirements categorization.



3.4 Quality plans

331. For the purpose of designing and implementing systems important to safety and any modifications to such systems, a quality plan specific to each individual system shall be prepared and adopted. However, the same quality plan may be utilised for several systems if the quality objectives, the methods for attaining the quality objectives and the organisation implementing the plan are the same for all the systems concerned.

332. The quality plan shall present

1. the organisation designing the system, complete with responsibilities and interfaces to other organisations involved in design;
2. the standards and guidelines, including the YVL Guides, to be applied in the design and implementation;
3. the stages of the design and implementation process;
4. the documents, records and other stage inputs serving as input data for each design stage;
5. the documents, records and other stage outputs created as an outcome of each design stage;
6. the stage reviews upon completion of individual stages including the timing, content and performer of the stage review, acceptance criteria, and the applicable decision-making procedures and responsibilities;
7. the procedures used in the supervision of subcontractors;
8. configuration and change management and procedures for product identification;
9. the management of conformity, design changes, and management of non-conformities;
10. the support processes utilised concurrently with design and implementation, complete with the associated management and quality procedures;
11. the division of responsibilities for the processes and decision-making procedures, including the procedures for modifying the quality plan.

333. The system-specific quality plan shall be prepared and implemented in compliance with the requirements set out in this YVL Guide and an applicable standard.

334. When standards-compliant processes and the quality manual of the design organisation are used, a detailed description of the application of the processes and guidelines shall be provided in the quality plan.

335. The requirements for the quality plan that complements the supplier's management system included in the delivery are set out in Guide YVL A.3.

3.5 Requirement specifications

336. The requirements concerning systems important to safety of the nuclear facility shall be defined to such a level of detail that a designer independent of the requirement specification process is able to carry out the re-design required for the in-service maintenance of the system its components as well as their modifications throughout the life cycle of the facility.

337. Requirements that are not considered functional requirements, such as the applicable quality requirements and standards, shall also be specified.

338. The applicability of the referenced standards and guidelines shall be justified. If an exception is made to a specified standard or guideline, such a departure shall be justified and its effect assessed.

339. The requirement specifications shall be unambiguous, consistent and traceable. It shall be possible to verify the fulfilment of the requirements.

340. The accuracy, completeness and consistency of the requirement specification of systems important to safety shall be assessed by experts who are independent of the design and implementation process. The assessment report shall present the observations made as well as a justified conclusion.

341. The traceability of the requirements in the various design stages shall be demonstrable. The traceability of the requirements in the various design stages shall be demonstrated as part of the qualification.

Figure 24 A page of the YVL document (STUK, 2013a) and the individual YVL document elements in a PDF (Adapted by the author from Karstila, 2013)

The initial classified dataset consisted of 2199 requirements from 10 YVL Guides in total, and was randomly split into training and testing datasets with the related labels. The arbitrary segregation was performed by a data scientist at Selko, once Fortum provided the classified dataset. This action is also emphasized in Figure 28 in the following subchapter. Two thirds of the requirements (1466) were utilized as training data and the rest (733) as test data. Hence, requirements of the same topic were utilized for both training and testing, but the actual requirements used for each stage were different. The second test dataset, so called blind test dataset, which is considered verification of the algorithm, was performed with unlabeled data of YVL Guide B.2 consisting of 65 paragraphs in total. Thereafter, the second blind test was employed using the requirements issued by the UK authority. The datasets and their amount of requirements are tabulated in Table 11.

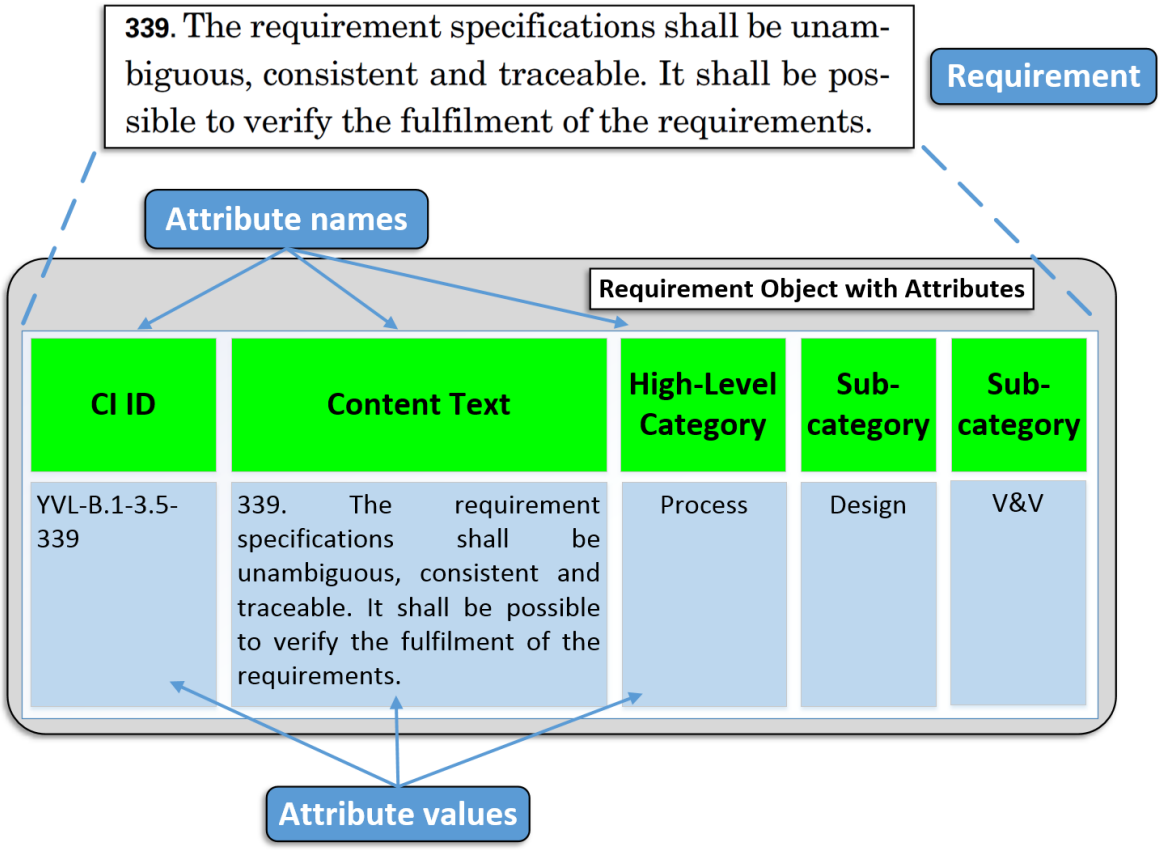


Figure 25 Requirement object with attributes (Adapted by the author from Karstila, 2013)

It is emphasized that the correct classes of the requirements in the test datasets have not been used for training the model even though it could practically be possible, at least after the usage in the testing. For this reason, the model has not seen the ground truths. The model learns a specific way to categorize requirements based on the discovered consistency, while processing the training data including the correct answers. Accordingly, the model classifies requirements based on the learned logic.

Table 11 Datasets and the number of requirements in each set

Dataset	Amount of Requirements
Classified Data	2199
Training Data	1466
Testing Data	733
Blind Testing Data (YVL B.2)	65
Blind Testing Data (UK SAP)	317

The second blind test, also listed in Table 10, was performed with the UK regulations which differ from the YVL Guides. The requirements set by the Office for Nuclear Regulation (ONR) were initially extracted from a PDF document which is available online on the website of ONR (ONR, 2014). The extraction was due to the lack of the SAP requirements in the Fortum database. It is emphasized that there are individually numbered paragraphs associated with the separately numbered principles presented in the boxes. However, to condense the scope of the study, only these principles were analyzed and referred to requirements. As indicated in Table 11, these principles amount to 317 unique requirements. The structure of the UK dataset in the PDF format is presented in Figure 26, whereas the structure of the Excel spreadsheet resembles the object with attributes illustrated earlier in Figure 23. There are similarities between the UK and YVL requirements recognized through the analyses. However, they are divergent enough to be compared with each other, and the UK set to be classified by the trained model.

ONR states that the numbering of principles is of the form XY.1 or XYZ.1 where the letters represent the thematic headings (ONR, 2014). There is no definition of the middle box available. Therefore, in the scope of this study, it is defined as a topic which only provides further information about the nature of the principle, and hence, employed to advance the manual categorization task. However, only a requirement ID and the corresponding requirement text are fed to the classifier, and the linkage retained during the classification of the model.

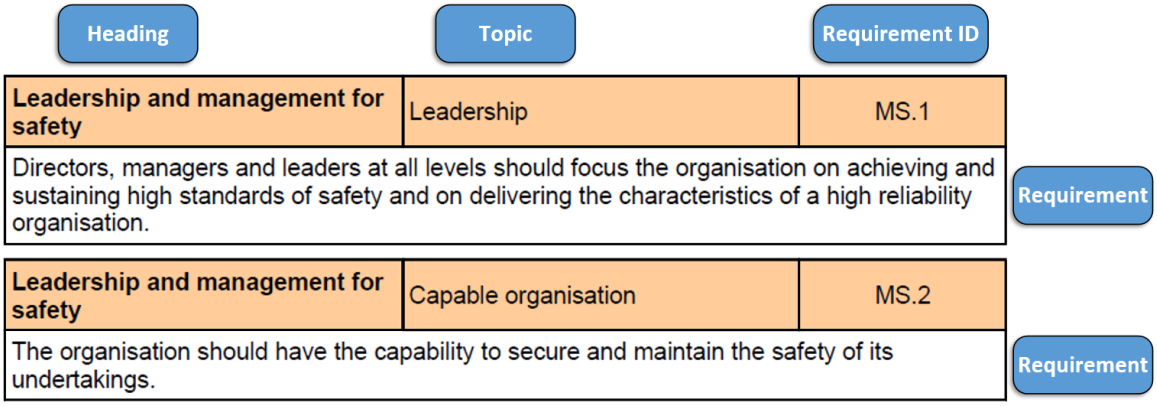


Figure 26 The principles issued by ONR are presented in boxes and separately numbered (ONR, 2014)

3.2 Methods

The essential actions performed in the case study are briefly discussed in this chapter. The aim is to provide a general understanding about the activities. The categorization process began with deciding the categories for the requirements, and determining the hierarchy of the algorithm. In addition, an initial meeting was held with the project team, in which a common execution plan and the responsibilities of the project were confirmed. A communication platform was created in order to have an efficient communication between the companies. The first steps of this development project are represented in Figure 27. The project was initialized by defining the objectives mentioned in Subchapter 1.2. The definition of those objectives clarified the task, the form and the required amount of data. The objectives were harmonized in the initial meeting with Selko.

The initial data (the YVL Guides) necessary for the study were exported from the Fortum database since they were already in the form of Excel sheets, yet being publicly available online in the form of PDF files (STUK, 2018a). Suitable YVL Guides were chosen to be used as training data based on the experience and knowledge about their content. The knowledge enabled to easily choose the relevant guides as it was essential to have guides which included both process and technical related requirements. The understanding of the data was the responsibility of Fortum, but also understanding the nature and basics of the data was required from Selko. Thereafter, Fortum analyzed and classified the requirements and provided them to be used in both training and testing. Thereafter, Selko pre-processed the data and randomly split the classified dataset into the training and testing datasets.

Figure 27 Beginning of the data science process

Many experts listed in Table 12 (Chapter 3.3) were consulted to understand and pre-process the data. Both common and individual meetings were held depending on the needs and nature of a subject. Furthermore, STUK VAHA-A categorization, which is explained in the following paragraph, was utilized to facilitate the task of attributing each requirement. However, it should be emphasized that the exact attributes utilized by STUK and this research are dissimilar. However, the general characteristics are similar.

VAHA-A was a requirements categorization project led by the Finnish Radiation and Nuclear Safety Authority (STUK). The aim was to classify all public requirements written in the YVL Guides, thus, amplifying the requirements specification. Performed in close cooperation with STUK, the Finnish licensees and license applicants, the project utilized specific attributes defined by the working and steering groups in the classification task. The established documents are not available online, but may be asked from STUK (STUK, 2018c).

Table 18 in Subchapter 4.2 indicates the time-consuming tasks needed for reviewing all the classified requirements. As shown in Table 8, the content of the guides utilized varies significantly from each other. The requirements were analyzed based on experts' perception and experience. One crucial matter was to canvass the context of each requirement from a lifecycle perspective, that is, at which stage would a requirement be implemented and in which way. Thereafter, it was more straightforward to assign correct categories for the requirements according to the developed hierarchy.

The categorization of each requirement set was verified in cooperation with the experts from the field in question. These experts, who have supported in the classification task, are tabulated in Table 12 in the following subchapter. Because the utilized hierarchy was a simplified version of the reality, some specific subjects of the requirements were included in a broader class. Furthermore, certain subjects were not considerably covered due to the finite amount of data and time available. However, the categorization was attempted to be logically performed.

The amount of the guides included in the training dataset was increased compared to the initial plan, as some categories lacked labels. Initially, the suggestion was to include 500 labels for each high-level class and 100 labels for each subcategory. In this study, a label is equivalent to a category or class, thus, being alternately used. The word "label" is applied to represent the context of machine learning, while the words "category" and "class" are used in the context of systems engineering, especially requirements analysis.

In general, there are three phases when building a machine learning model, specifically collecting the data, training the model, and testing it. The latter one refers to the verification and validation of the model. There is also a different execution time for each of these steps. Furthermore, the test phase can be divided into subtasks of the results and their validation as indicated. Once the requirements had been gathered, classified and verified, they were provided as a classified dataset illustrated in Figure 28. Using these requirements, the model was trained, tested and scored. In addition, totally new datasets were utilized to further test the algorithm. The results of the classifications were verified after each test to evaluate the model.

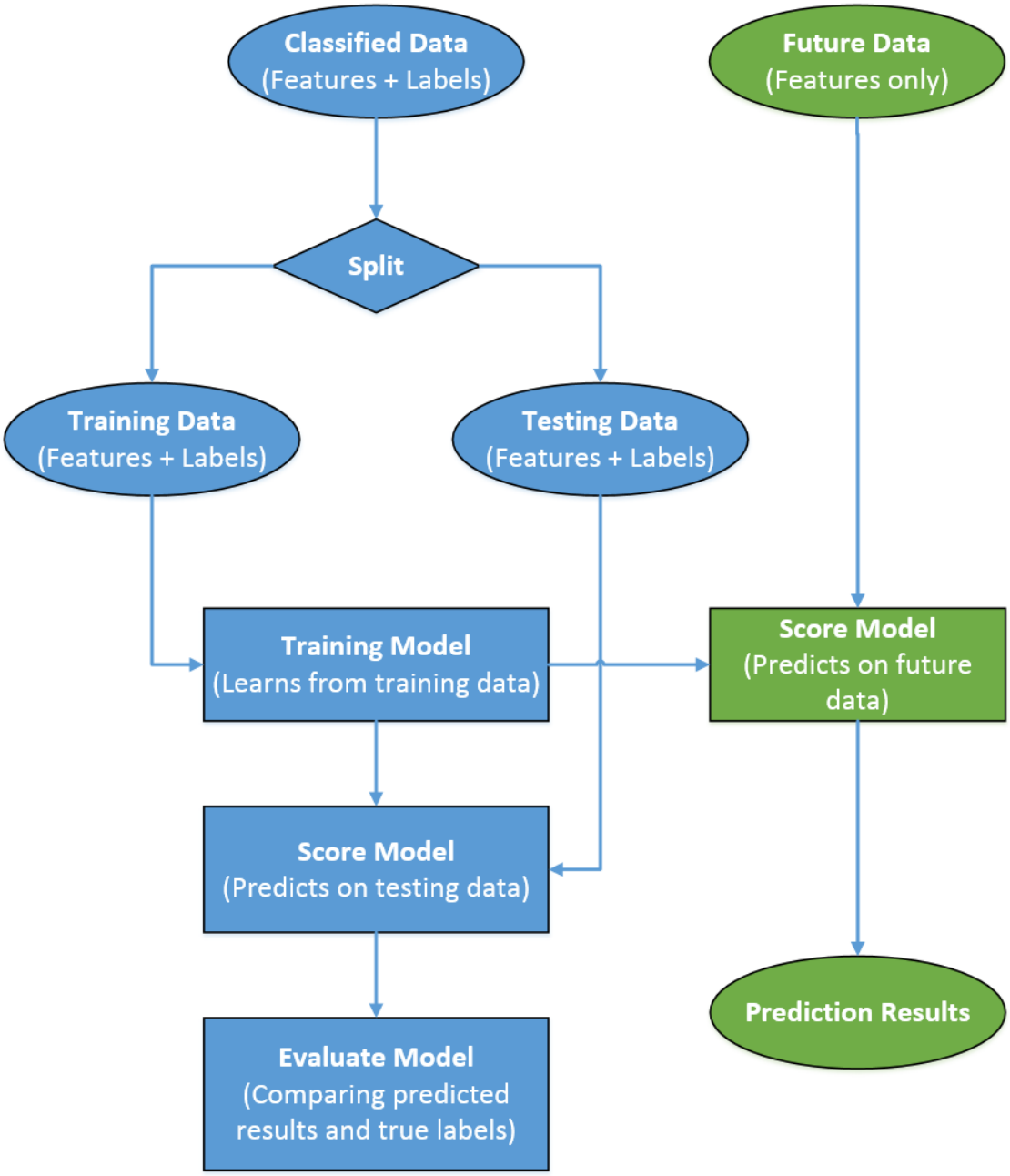


Figure 28 Data science process after pre-processing

3.3 Trustworthiness of Study

The trustworthiness of the study was improved with the help of the Fortum experts assisting the thesis worker responsible for the research. Their assistance enabled the proper understanding of the classified requirements, thus, ensuring as coherent classifications as possible. The categorized requirements used for training and testing were reviewed and verified by 10 Fortum experts. Their roles and specialties are listed in Table 12 below. It is emphasized that not every expert reviewed the whole data collection, but only the applicable parts.

Table 12 Fortum experts who have supported in the requirements categorization process

Job title	Specialty
Head of Nuclear Engineering	Nuclear Engineering
Technology Development Manager	Systems Engineering and Licensing
Automation Manager	I&C and Licensing
Electrical Design Manager	Electrical Engineering
Design Engineer	Requirements and Configuration Management
Design Engineer	Safety Design and Analysis
Design Engineer	Safety Design, Safety and Seismic Classification
Specialist	Strength Analysis
Specialist	Structural Engineering
Section Manager	Process Chemistry

After the review process carried out together with the experts, uncertain classifications and the related comments were collected for the further evaluation. Thereafter, the uncertain paragraphs were verified in separate meetings with the experts having the suitable competences. Given that the algorithm tries to learn the logic from the data, the verification of the labels was extremely important to ensure the consistency of data. In other words, the data should be systematically categorized to be consistent.

Because the classification of the requirements is always challenging and depends on individuals' perspectives and attitudes, a pre-determined requirements hierarchy and the content examples of each class were determined to improve the quality of the classifications. The hierarchy and content examples are further described in Chapter 4 Results.

After the reviews and verification of the classified dataset, the data were provided to Selko for training and testing the algorithm. At that stage, Selko had already a rudimentary code available, but the tailored algorithm was dependent on the needs of Fortum. Hence, it was extremely important to be conscious of the internal needs and properly communicate them to the service provider. In addition to training and evaluating the algorithm, Selko was also responsible for calculating the overall labeling accuracies and providing all the necessary information to support Fortum in the assessment of the model.

4 Results

This research generated many valuable results. These results include a specific hierarchy for the requirements classification with the pre-determined content for each class, both manually and automatically categorized requirements as well as a natural language processing algorithm including a domain language model and a classifier. Several text classification models using recurrent neural networks (RNNs), specifically, bidirectional RNNs composed of LSTM cells were explored for developing the language model. Finally, only one solution has been further examined to solve the multi-class classification problem, namely a feedforward neural network. Furthermore, the classification accuracies were attained while evaluating the machine learning algorithm on the categorization. These main outcomes are discussed in the following subchapters.

Initially, the intention was to develop an algorithm classifying requirements according to the certain hierarchy. To initiate the study, relevant features of the raw data were defined, that is, labels to represent the classes of interest. After the collection and classification of the data, it was ensured that the whole set to be used in training was appropriately categorized. Training was based on an iterative method called gradient descent to find good predictors by minimizing the average loss incurred for the labeled training data.

After the training, the algorithm was validated by entering a labeled dataset, so called validation data, which had been separated from the initial data. In the preliminary testing, the main concern was to detect and avoid overfitting. Once the algorithm was validated, it was tested on another dataset of which subject differed from the ones utilized in the training and validation. The phase is generally called a blind test, because only the features (requirement texts) are entered to the model. It was important to experiment the capability of the model to classify requirements of which subjects are slightly different to the ones in the training set.

Since the YVL Guides consist of descriptive-style requirements, one can explicate a requirement in many ways without necessarily being wrong. Furthermore, a requirement itself may include many other requirements and affect several different disciplines, leading to a multi-class categorization. The algorithm has to predict multiple labels informing that a requirement belongs to specific categories with a certain accuracy.

The manual classification of each requirement was the most time-consuming task in this research. However, the time consumed for the manual work was not recorded, but the estimation is approximately two months only including full working days. Thereafter, the categorizations were verified together with the experts mentioned in Table 12. Because reviewing the classifications was also a long-lasting process, the duration for verification of each requirement set is tabulated in Table 18 at the end of Chapter 4.2.

4.1 Requirements Hierarchy

A specific hierarchy for the classification was defined in this case study. The concept was to experiment the level at which an algorithm could recognize different hierarchies and hierarchy levels. However, one of the long-term objectives is to implement an algorithm to perform according to the ADLAS hierarchy and methodology. As the ADLAS hierarchy is based on the various levels, a requirement belonging to a certain category may also have an effect on a subcategory at the lower hierarchy level. Related to the hierarchy levels, the specific categories were defined to cover relevant systems engineering processes and subjects in the form of the *process* class divided into five subcategories emphasized in Figure 29. The high-level classes are *Heading*, *Process* (P), *Technical* (T) and *Reference*, whereas the subcategories include the following classes: *Design* (D), *Verification & Validation* (V&V), *Qualification* (Qf), *Documentation* (Do) and *Licensing* (Li). The abbreviations used by both the experts and the algorithm were adopted for the classification tasks to facilitate the categorization and the interpretation of the results. These abbreviations also facilitates the understanding of the classes demonstrated later in Chapter 4.2 Requirements Classification.

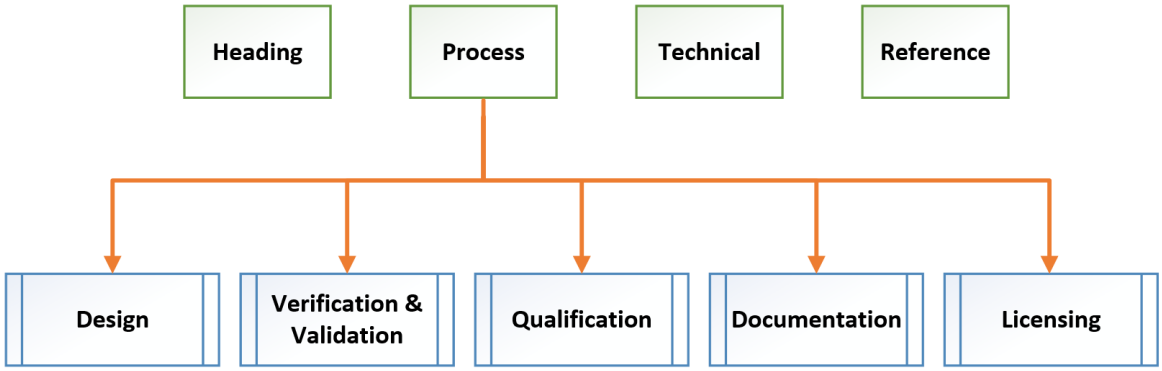


Figure 29 Hierarchy of the classification

Because the ADLAS hierarchy is based on technical requirements and the methodology is the intellectual property right of Fortum, the intention was not to provide the structure in terms of subcategories of the technical class. Overall, the current concept is considered representing the idea according to which an algorithm could be developed based on the ADLAS hierarchy. In contrast, the process class comprises the system lifecycle processes defined by ISO 15288, and presented earlier in Figure 6. As can be noticed, ISO separates processes into four subdivisions, namely *Technical processes*, *Technical management processes*, *Agreement processes* and *Organizational project-enabling processes* (ISO/IEC/IEEE, 2015). Furthermore, in the INCOSE Systems Engineering Handbook, the categories *Technical management processes* and *Organizational project-enabling processes* are termed as *Project processes* and *Enterprise processes,* respectively (INCOSE, 2015). In the context of this study, each of these classes belongs to the process category. Hence, it is worth noting that the analyzed technical requirements do not equal to the technical processes defined in the standard and shown in Figure 6. On the contrary, the technical requirements are considered both non-functional and functional product requirements.

Designing the particular requirements hierarchy involved many specialists with diversified competences in systems and requirements engineering. The outcome was based on a compromise between the different views of the experts. One of the most essential factor was to provide useful and practical hierarchy which would satisfy the need of separating different hierarchy levels from each other, and concurrently, include categories which can be easily discovered among the requirements. Especially, determining the subcategories of the process class involved special expertise and experience on the YVL Guides and requirements engineering.

The categorization of the requirements depends always on people carrying out the process, in particular, their points of view and purposes. The terms and categories were defined at the beginning of the categorization process. Additionally, reasonable and consistent understanding on the data was enabled. In the scope of this Master's thesis, most of the terms are defined according to international standards, such as ISO 9000 (SFS, 2015), ISO 15288 (ISO/IEC/IEEE, 2015), ISO 24748 (ISO/IEC/IEEE, 2016) and ISO 29148 (ISO/IEC/IEEE, 2011a). Certain definitions were improved, for instance, with examples. The following chapter discusses more about the classification of the requirements, and the definition or the content of the categories.

4.2 Requirements Classification

As mentioned in the previous chapter, the particular hierarchy represents only a specific case; that is, it is one way of classifying requirements. In addition, it is an intentionally simplified grouping for the stated purpose. The generic content of each category is explicated in Table 13. In the table, only certain examples of the content of the categories are provided to indicate main differences between the categories. The intention is not to explain the exact definition of each class due to two reasons: 1) it is rather complicated because of ambiguous requirements, and 2) the definitions include plenty of internal knowledge of Fortum. Therefore, the additional and formed knowledge is only desired to be internally managed.

Table 13 Generic content examples of each requirement category

Category	Content Examples
Heading	Titles defining the context of a chapter
Process (P)	System lifecycle processes (design, manufacturing, commissioning, etc.)
Technical (T)	Functional and non-functional product requirements, safety achieved through a component, system and/or architecture
Reference	Requirements informing relevant guides, descriptions
Design (D)	Design processes, quality, configuration and requirements management as well as operation
Verification & Validation (V&V)	Measurements, analyses and tests
Qualification (Qf)	Justifications, demonstrations, type approvals and type tests
Documentation (Do)	Records, deliverable matters, data, results, licensing materials
Licensing (Li)	Authority (e.g., STUK) involvement

A requirement may belong to several classes due to the ambiguity of the requirements. In other words, a certain requirement can be labeled both process and technical if it refers to both subjects. This situation is intractable since various requirements have to be explicated. Additionally, the analysis may differently be performed depending on the interpretation and the context. As stated in the previous chapter, the distinction between the process and technical classes is illustrated by the system lifecycle processes and products related requirements. For instance, the system lifecycle involves agreement processes which can be considered licensing processes in the nuclear industry, or many quality assurance tasks to ensure the adequate quality.

Table 14 lists examples of the recognized ambiguous requirements. Regarding the first requirement 518, it is not necessarily explicit whether “a programme” means a computer program, a procedure or something else. The second example includes both process and technical issues as it begins with requiring the usage of testing and analyses, but finally demands that there shall be no unnecessary function, which refers to a technical requirement. The final object is an excellent example of a requirement containing nebulous words or terms of which meaning has to be delineated prior to continuing the elicitation task. In this case, there are words, such as “special care” and “minimise”, which have to be separately defined, for instance, according to a process predetermined in the quality documentation. It is underlined that these challenges in the interpretation of requirements are not limited to any

authority, but also involve subjects, such as contracts and international demands. Similar challenges from the field of genre classification have been recognized as the data include different preferred vocabularies and various writing styles even for documents within one genre. This means the data are oftentimes heterogeneous (Ikonomakis *et al.*, 2005).

Table 14 Examples of ambiguous requirements (STUK, 2013d, 2013g, 2013a)

YVL Requirement	Description
YVL-B.5-5.3-518	518. A sampling programme shall be in place for monitoring the chemistry parameters and activity concentrations in the primary and secondary circuit.
YVL-E.7-5.3-517	517. Testing and analyses shall be used to ensure that the electrical or I&C systems or equipment in safety class 2 contain no unnecessary functions that could be detrimental to safety.
YVL-B.1-4.1-408	408. […] In particular, special care shall be taken in the choice of materials in order to minimise the future quantities of radioactive waste to the extent practicable, and to facilitate decontamination. […]

Examples of the paragraphs belonging to heading and reference classes are provided in Table 15, whereas examples of the requirements belonging to process and technical categories are presented in Table 16. In the scope of this study, differences between these two categories may be easily recognized as process requirements relate to both systems engineering lifecycle processes identified in Chapter 2.1.2 and the definition of a process as stated in ISO 9000:2015 Standard (SFS, 2015). In contrast, technical requirements relate to the product requirements which can be divided into functional and non-functional requirements described in Chapter 2.1.4 (Hull *et al.*, 2011; INCOSE, 2015). Therefore, they are included in the technical category. An example of a product is a system, and requirements related to the system are considered technical requirements. The technical requirements may, for instance, state the function or structure of the products.

Table 15 Examples of the heading and reference paragraphs (STUK, 2013d, 2013g)

YVL Requirement	Description	Requirement Type
YVL-B.5-4	4 Pressure control of the primary and secondary circuit	Heading
YVL-B.5-4.1-403	403. More detailed instructions for the design of valves of nuclear facilities are given in Guide YVL E.8.	Reference
YVL-E.7-3.1-310	310. The design, manufacture and testing of electrical equipment and cables in safety class 3 other than those listed in para. 309 shall employ applicable Finnish or international electrical equipment standards.	Process & Reference

Table 16 Examples of the process and technical requirements (STUK, 2013a)

YVL Requirement	Description	Requirement Type
YVL-B.1-3.2-316	316. The organisations involved in the design shall have capable processes in place for managing requirements.	Process
YVL-B.1-3.5-341	341. The traceability of the requirements in the various design stages shall be demonstrable. The traceability of the requirements in the various design stages shall be demonstrated as part of the qualification.	Process
YVL-B.1-5.1-5103	5103. The reactor cooling system and the associated auxiliary, control and protection systems shall be so designed as to ensure that the design parameters of the reactor primary circuit are not exceeded in operational states.	Technical
YVL-B.1-5.2.6-5239	5239. A single failure occurring in a nuclear power plant's I&C system must not cause an initiating event that exceeds the severity level of an anticipated operational occurrence.	Technical

The process requirements are automatically associated with at least one subcategory. Therefore, according to the defined setup, there cannot be a process requirement without any subcategory associated with it in the classified dataset. The examples of the process requirements belonging to some of the defined subcategories are presented in Table 17. It further emphasizes the difference between process and technical requirements: the process requirements concern the ways of manufacturing a product but not the product itself, whereas the technical requirements apply to quality and characters of the product. In terms of requirements engineering in nuclear power industry, the process requirements are considered more time-critical than the technical requirements, because they involve procedures and processes to be in place prior to initiating any manufacturing activity.

Table 17 Examples of the process requirements including the associated subcategories (STUK, 2013a, 2013g)

YVL Requirement	Description	Process Subcategories
YVL-B.1-3.3-320	320. The configuration management processes and procedures shall cover the entire lifecycle of the facility, from design to commissioning and operation.	Design
YVL-E.7-3.4.2-356	356. If the component in question is not serially manufactured, a summary of the factory test results shall be presented together with the final suitability analysis.	Qualification, Documentation
YVL-B.1-5.2.7-5254	5254. Factory tests shall be carried out in a factory acceptance test environment designed for testing purposes.	Verification & Validation

The initial dataset consisted of total 2199 requirements. They were labeled according to the hierarchy shown in Figure 29. The distribution of the high-level labels is illustrated in Figure 30. It should be noticed that the total number of labels is more than the total amount of requirements because a requirement can concurrently belong to several classes. For instance, a requirement may concern process and technical aspects, thus, being associated with both categories.

Titles are exceptions because they can only belong to one class, namely the class heading. The process category includes approximately three hundreds requirements more than the others because the process requirements are typically considered ambiguous. Furthermore, especially the minimum number of qualification requirements was challenging to be satisfied, resulting in gathering additional process requirements. They slightly differ from the V&V requirements, and usually involve other classes, as well. A practical example of emphasizing the V&V aspects is as follows: verification means checking that the power plant was finally built as defined in the design phase, whereas, validation means ensuring that the plant is suitable for the pre-defined purpose, that is, safely generating electricity.

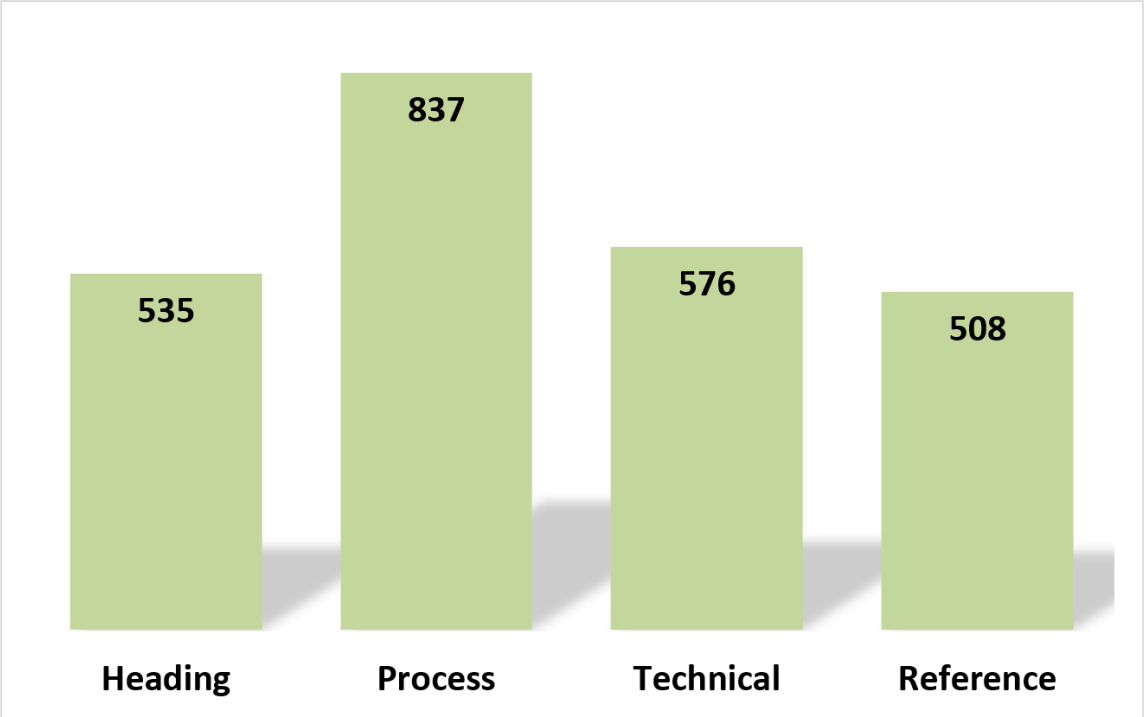


Figure 30 Number of the high-level categories in the initial dataset

The total number of the labeled subcategories is 1328 which includes many requirements associated with several labels. Accordingly, as the process class involves at least one subcategory, the process requirements may have been classified into more than one class. The distribution of subcategories is depicted in Figure 31. The design and documentation classes are distinct, mainly because documentation related requirements also involve other subjects, such as communication with the authority as well as quality, configuration and requirements management related topics. Furthermore, in this case study, the design class contains a wide range of subjects, such as quality, procurement, standards and manufacturing. A convenient example of a process requirement involving a documentation task is a quality claim obligating that quality management system procedures and processes

shall be designed, documented and kept updated. The charts presented in Figure 30 and Figure 31 can be interpreted as follows: the total number of requirements belonging to at least the process class equals 837 requirements, and those requirements have been further classified resulting in the distribution of sub-classes, in which process requirements may have been categorized into several classes.

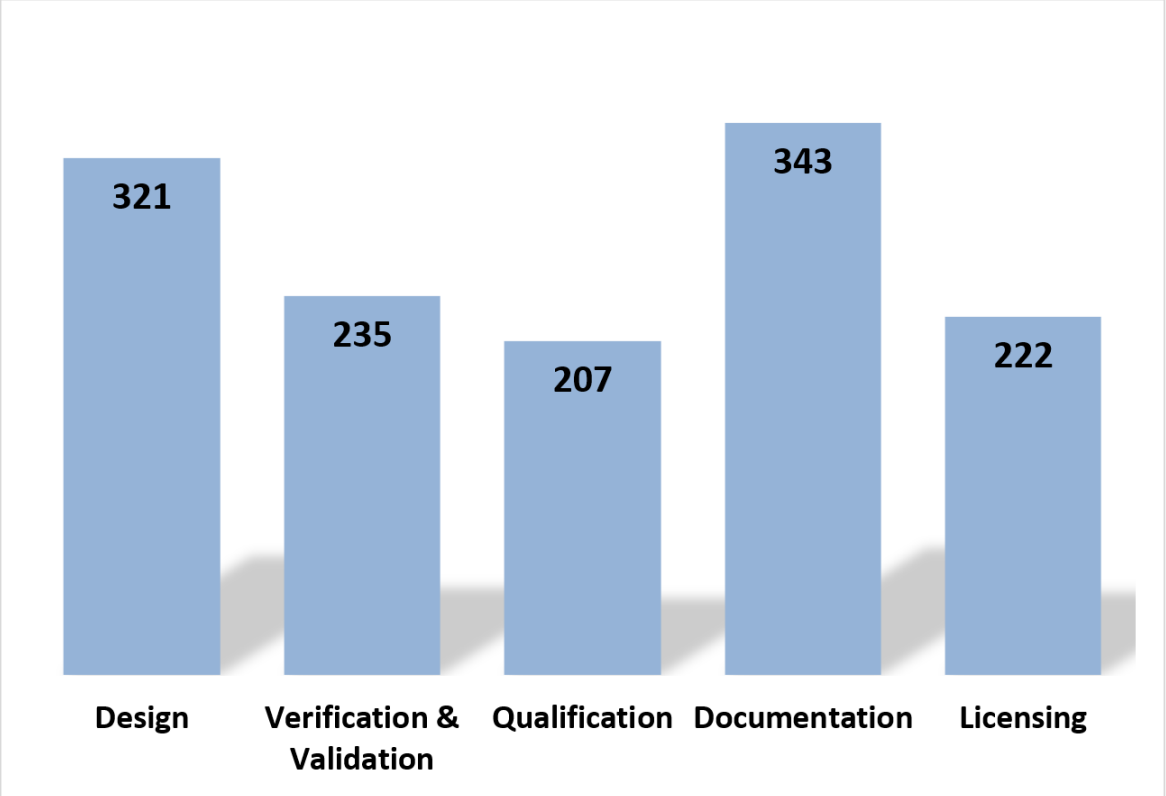


Figure 31 Number of the subcategories in the initial dataset

As mentioned earlier, the distribution of the sub-requirements is due to the multi-label case, in which not only a requirement can belong to several high-level classes but also to many sub-classes. The amount of the requirements in the classes *design* and *documentation* separate themselves from others mainly due to the nature of the requirements. The requirements usually include subjects which have particularly been defined belonging to these classes. To be precise, in case a requirement belonging to the class *qualification* is desired to be classified and included in the dataset, the same paragraph may also include requirements to be allocated to other classes. For instance, a qualification requirement oftentimes involves documentation tasks which are also included in other paragraphs associated with different classes.

The requirements were not only classified for the initial classified dataset, but also for the blind tests. Even though the blind tests lacked all the labels when testing the model, ground truths (i.e., correct answers) were manually and independently determined, thus, avoiding any influence of knowing the other result in advance. Therefore, there was no suggestion provided by the algorithm during the manual requirements categorization. Hence, the results of the model could be checked against the classifications determined by the experts. The only exception in terms of knowing the results of the machine is the first blind test, in which

one-third of the requirements were finally labeled with the help of the results of the model. This is discussed more in Chapter 4.4 Algorithm Classification. However, each requirement and the related classes were always verified with specific experts. Their job title and specialties were listed in Table 12 in Chapter 3.3 Trustworthiness of Study. The review meetings of which the thesis worker was in charge were always organized with the relevant internal experts.

Not only were the classification and verification processes time-consuming, but also they were complex. Due to the amount of the arranged meetings, Table 18 lists each requirement set (i.e., a YVL Guide) and the time consumed to verify the related categorization. The relevance of each label was analyzed and agreed in the meetings, thus, utilizing valuable knowledge of the internal experts to have the coherent requirement sets.

In practice, the thesis worker was responsible for categorizing each requirement. Therefore, a raw classification of the requirements was available when meeting with the experts, who usually had their own suggestions at least for uncertain classifications prior to the meeting. However, the time used for individually categorizing requirements was not documented. For this reason, Table 18 only tabulates the time in hours consumed for the common meetings aiming at verifying the classes associated with each requirement. The overall time equals to approximately 75 hours. It should be emphasized that the classification of the requirements is generally a time-consuming process, given that the requirements utilized in the case study are difficult to understand and usually contain many other demands; that is, the requirements are ambiguous.

Table 18 The time consumed to verify the categorization of each requirement set

Requirement Set	**Time (h)**
YVL B.1 Safety Design of a Nuclear Power Plant	8
YVL B.2 Classification of Systems, Structures and Components of a Nuclear Facility	4
YVL B.4 Nuclear Fuel and Reactor	5
YVL B.5 Reactor Coolant Circuit of a Nuclear Power Plant	7
YVL E.3 Pressure Vessels and Piping of a Nuclear Facility	7
YVL E.4 Strength Analyses of Nuclear Power Plant Pressure Equipment	5
YVL E.6 Buildings and Structures of a Nuclear Facility	6
YVL E.7 Electrical and I&C Equipment of a Nuclear Facility	8
YVL E.8 Valves of a Nuclear Facility	6
YVL E.9 Pumps of a Nuclear Facility	5
YVL E.10 Emergency Power Supplies of a Nuclear Facility	6
UK SAP Safety Assessment Principles for Nuclear Facilities	8
TOTAL	**75**

4.3 Natural Language Processing Algorithm

In addition to the classified and consistent data, the importance of a well-designed and proper algorithm is evident. Together with the relevant theory, this chapter aims to provide all the necessary information required to understand the basics of the functions of the algorithm. The deep learning architecture of the natural language processing (NLP) algorithm is extremely complex including a substantial amount of weights and biases to be trained and adjusted within the entirety. Hence, the related calculations are computationally intensive. This chapter discusses the most relevant structures and actions related to the development of the particular NLP algorithm. However, the descriptions are simplified due to the complexity and the intention to constrain the scope of the thesis.

Initially, Selko had a basic structure for the model. The algorithm was customized to suit the case study in accordance with the needs of Fortum. The development of the model included several meetings between Fortum and Selko to ensure that both parties have the same expectations. Overall, the common meetings continued approximately 20 hours. Eventually, training the model with the classified data enabled the algorithm to learn a specific issue in a certain manner. Thus, the multi-class classifier predicts a class, among a set of classes, to which the particular requirement may belong.

Once the classified dataset, including the features and the related labels, were available, it was supplied to Selko for the training part. The classified and verified dataset included all the 2199 categorized requirements in an Excel spreadsheet. The "Classified Data" part in Figure 28 represents this stage. Prior to starting the development of the model, the balance of the data was checked to ensure equal amount of requirements covering each class. Furthermore, the most frequent words per label were analyzed to understand the data, and a scatter plot of the feature vectors was plotted to ensure that the classes were linearly separable. However, the classes were not linearly separable, thus, requiring a complex model. The development task was proceeded once everything was acceptable.

The general workflow is presented in Figure 32. The main idea is to provide text in the form of vectors and train the recurrent neural network (RNN) with long short-term memory (LSTM) cells to attain the language model. The LSTM cell was briefly discussed in Chapter 2.3.3 Natural Language Processing. The language model of which two main building blocks are illustrated in Figure 33, provides feature vectors which are entered to the feedforward neural network (FNN). Both the RNN and FFN are trained by adjusting the weights and biases of the networks. These adjustments are due to backpropagation. In the language model, the network learns to predict next words given the context, whereas, the FNN learns to predict a correct class to which the requirement could belong. The training of the FFN is based on providing the correct classes associated with each requirement in the training dataset. Finally, the output is provided, that is, predicted classes for each requirement.

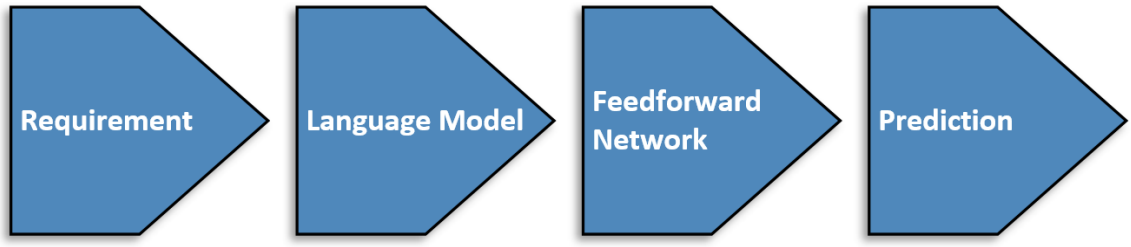


Figure 32 General workflow for both the training and testing phases

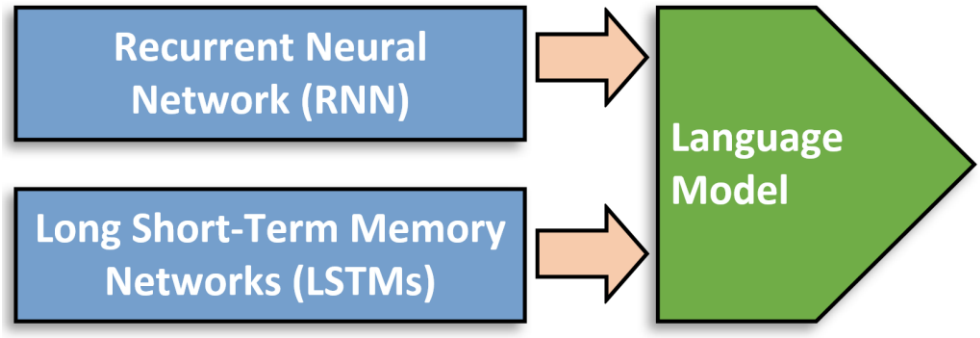


Figure 33 Language model comprises LSTM cells having the structure of RNN

The overall workflow for training the classifier is presented below in Figure 34. The figure emphasizes the involved three main datasets and the product generated after the utilization of each dataset. The two primary datasets were used to train the Fortum language model through transfer learning, and the manually labeled training dataset was provided to train the classification task. Once the Fortum classifier had been trained, the requirements could be fed to the classifier, which depending on the ability to recognize them, assigns the labels for each requirement. In the ideal situation, the requirements should be similar to each other in order to facilitate the categorization task of the algorithm.

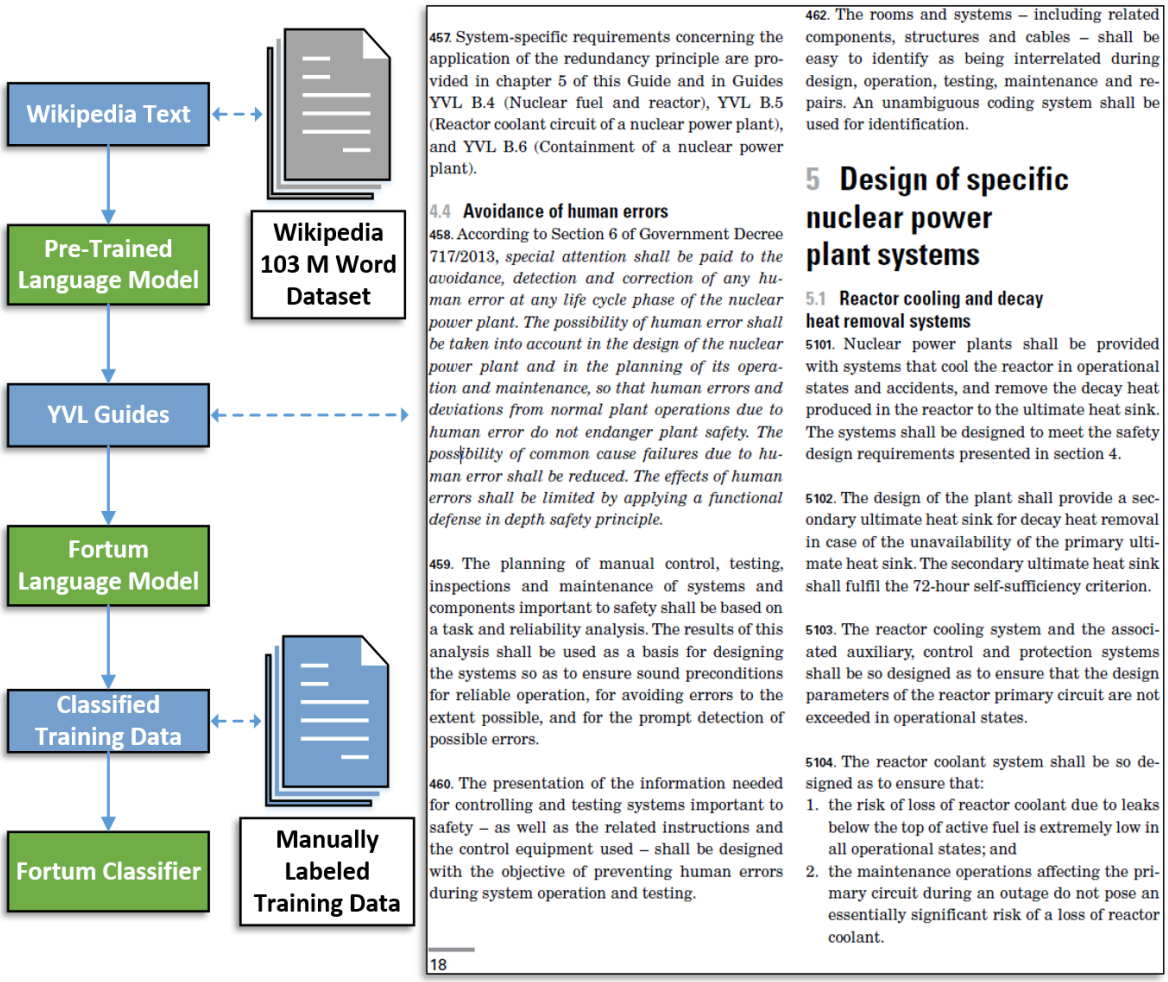


Figure 34 The overall workflow for training the Fortum classifier

The architecture of the language model is the RNN with three LSTM layers, each comprising of 70 LSTM cells which totals 210 cells. Figure 35 is a simplified version of the real architecture of the LSTM network, but still emphasizes the structure including all the main building blocks as well as interdependences. It is emphasized that the structure of the language model remains the same. The information flows horizontally only from left to right and vertically from bottom to top. However, the algorithm has bidirectional provision. The normal LSTMs were employed because they provided the best results when compared with the bidirectional LSTMs (biLSTMs). In other words, it was a design choice based on the accuracies. Given that there are two times more weights in the biLSTMs, the model would also require more labels, that is, training data. The RNN code was based on an open source code provided by PyTorch (Torch Contributors, 2018). The functions are briefly introduced on the following page.

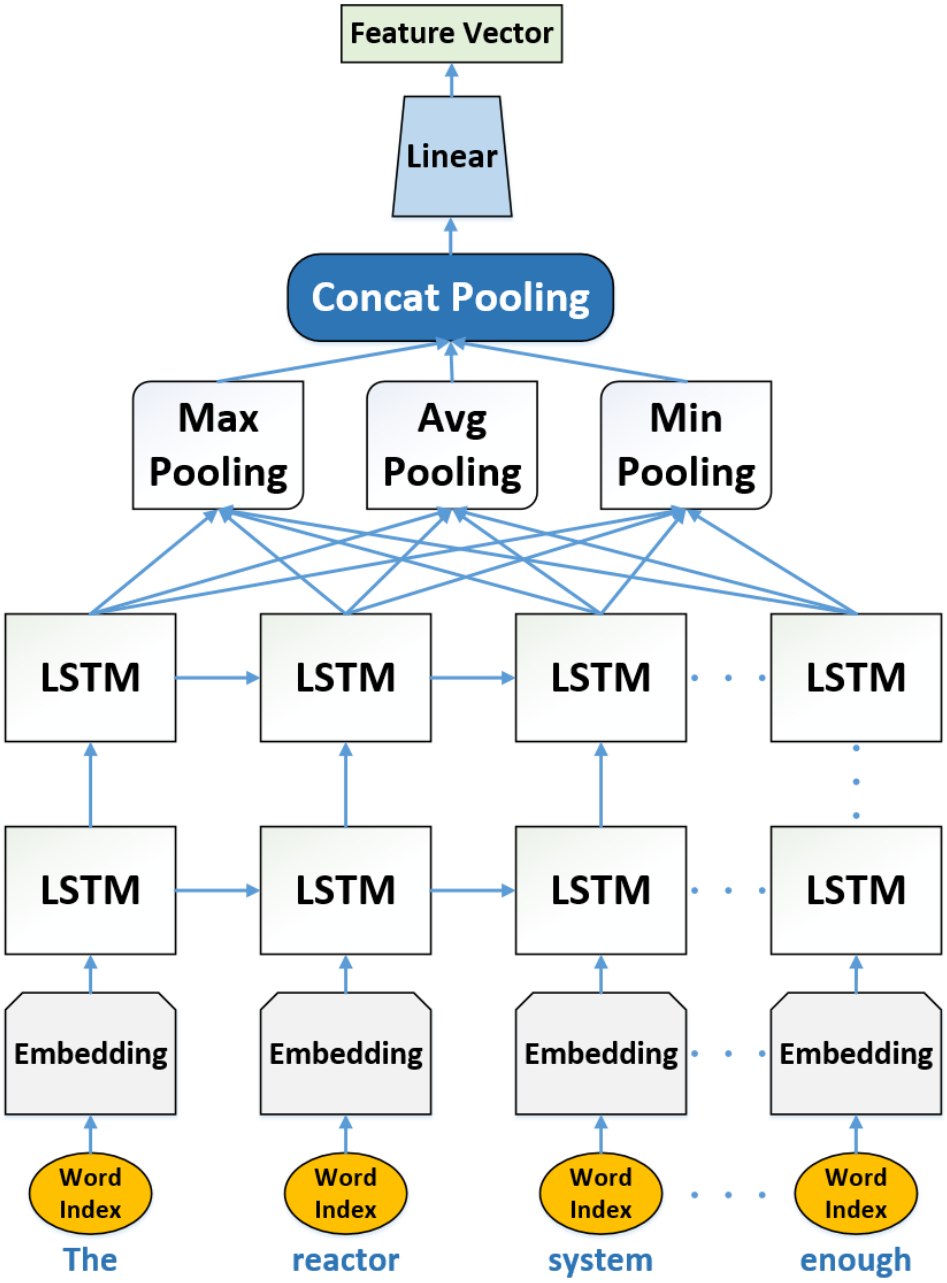


Figure 35 An illustrative architecture of the language model

Characteristically for deep neural networks, they require enormously data, and yet there are usually low quality data. Based on past experience, approximately 10,000 labels for each class would have been required to have a workable classifier. For this reason, transfer learning with LSTMs were adopted to alleviate the problem, thus, avoiding the sheer volume of labels.

Initially, a sentence is provided for the classification task, in which LSTMs model the sentence as a sequence. Every word and the corresponding embedding is examined, followed by feeding the embeddings as input into the RNN with LSTMs. As a design choice, the word embeddings (x_t) are 400-dimensional vectors, and the first two layers output hidden states (h_t) of 1150 dimensions. Instead of providing the equivalent vectors, the LSTM cells of the last layer output hidden vectors of length 400 each, which undergo maximum, minimum and average pooling operations. This activity corresponds to the feature transformation described in Figure 20 in Chapter 2.3.3 Natural Language Processing. As a result, there are three 400-dimensional vectors. These vectors are concatenated to result in a 1200-dimensional vector which is used as an input to the feedforward network. In other words, the embeddings of the current words are converted into the vectors corresponding to the words, generated by the language model, and fed into the FFN as input.

As briefly described in Chapter 2.3.3, each LSTM cell tries to forget, select and create a new memory. The cell involves many matrix multiplications and linear algebra, mostly in the form of neural networks corresponding to each of these three functions. It is highlighted that even the weights and biases, which are shared within every block and associated with these networks, are adjusted during the training. However, every cell has its own weights. Furthermore, it is noted that sharing the weights is not arbitrary, but relies on the design choices and experiment. Understanding the behavior of the network is crucial when determining the potential allowance for distributing the weights.

The first language model was initially built by using the Wikipedia data of 28,595 preprocessed English articles, corresponding to approximately 103 million words. This was considered the most convenient way to start with due to the sheer volume of data available for free. In addition, the data profoundly represented the English language. When the model was trained to understand the texts in Wikipedia, it was considered the Wikipedia language model or pre-trained language model. The main training phases of the language model are represented in Figure 36.

The words in the Wikipedia dataset are converted into vectors, that is, random word embeddings. These initial embeddings are fed into the LSTM network which is an RNN composed of LSTM units. The network learns to predict the next word given the previous ones, and finally both the weights of the network as well as the embeddings are adjusted by training. Thereafter, the learned word embeddings include the semantic meanings of the Wikipedia words following by the trained network, which can understand the similar text by predicting the words in a sentence. This prediction task converts the random embeddings to the learned embeddings representing the language model. Initially, term frequency-inverse document frequency (TFIDF) and singular value decomposition (SVD) were used to obtain the feature vectors later utilized with the extended gradient boosting (XGBoost) algorithm for categorization.

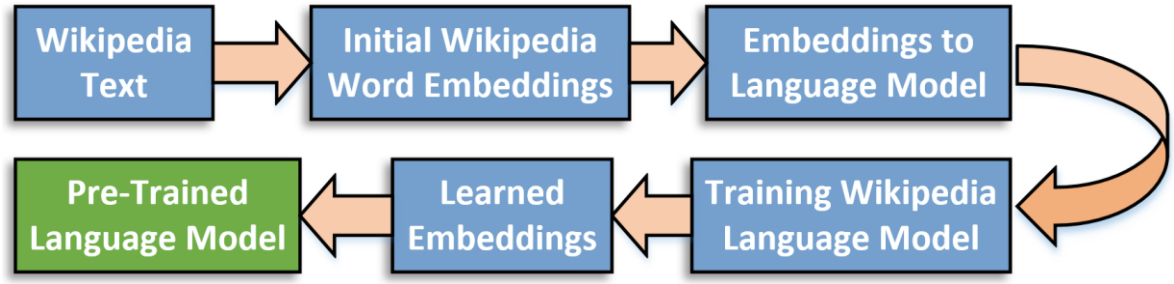


Figure 36 Training of the Wikipedia language model

The weights of the three LSTM layers were initialized with the model trained on the Wikipedia text dataset, followed by fine-tuning the weights with the domain-specific text data. In other words, the model was later improved with the YVL Guides B.1, B.2 and B.4, resulting in a customized language model familiar with the domain language, that is, the language of the YVL Guides. Thereafter, the model especially understands the text written similarly to the guides. Figure 37 represents the workflow of training the Fortum language model. The training was begun with the word embeddings from the Wikipedia model. Due to the sheer volume of the Wikipedia words in English, the words are mostly the same as in the YVL Guides. Only the semantic meaning is fine-tuned with the domain language and unfamiliar words of which vectors are first random, but adjusted once the words are fed into the LSTM network. Thereafter, the weights of the RNN are changed to correctly predict a word given the previous words in the requirements, that is, the context. The training of the language model occurs by backpropagating the error, thus, being similar to the training of the FNN. Hence, the specific language model is able to recognize sentences in the given domain.

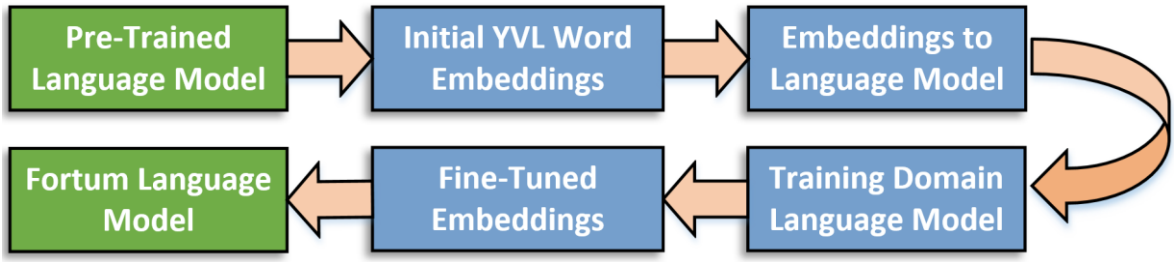


Figure 37 Training of the Fortum language model

The maximum length of the sentences were initially studied to perceive the maximum amount of words in the requirements. Due to the static input of 70 LSTM cells in each layer, 70 input examples (i.e., words) can be fed to the language model. In the case of the shorter requirements, the rest embeddings are called “unknown embeddings”, that is, each component of the embedding is zero. Thus, the training of the language model takes into account the entire requirement.

The Fortum classifier consists of two main parts, specifically the trained feedforward network and domain-specific language model, as illustrated in Figure 38. They are tightly connected with each other because the language model provides information of the text for the feedforward network which again performs the prediction task. The 9-dimensional output layer of the FNN corresponds to the probabilities of the pre-determined categories which are shown in Figure 39.

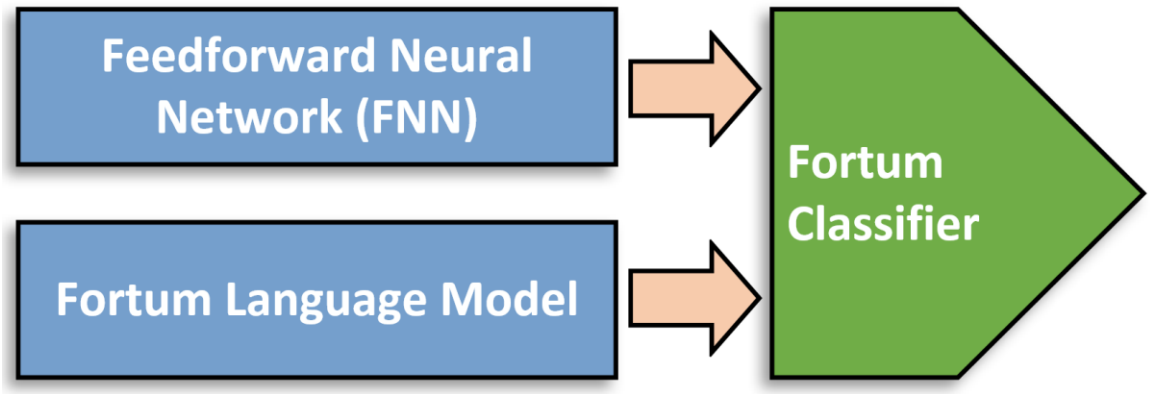


Figure 38 The Fortum classifier consists of the FFN and the domain-specific language model

The classified training data, or more precisely the requirements, are fed into the Fortum language model. Then, the next word is not predicted anymore, but the end of the network is replaced with a feature vector. The language model having the structure as described in Figure 35 outputs the feature vector of 1200 dimensions. The feature vector representing the whole requirement is fed as an input into the feedforward network of which architecture is emphasized in Figure 39. The network consists of an input layer of 1200 dimensions (i.e., nodes), one hidden layer comprising of 50 neurons, and a 9-dimensionsal output layer. All nodes are fully connected and the information flows straight from the input layer to output through the hidden layer between them. The respective order of the training phase is presented in Figure 40 illuminating the training workflow of the classifier.

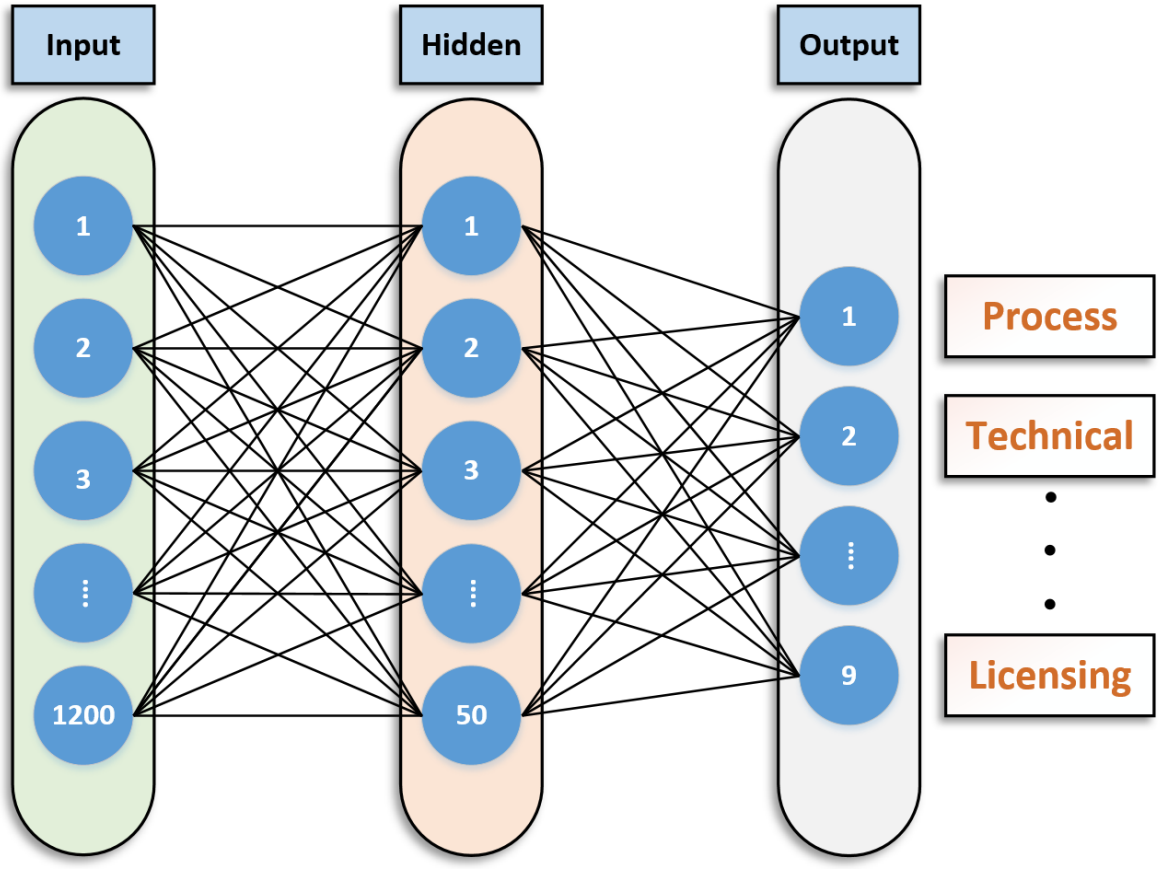


Figure 39 An illustrative architecture of the feedforward neural network utilized in the classifier

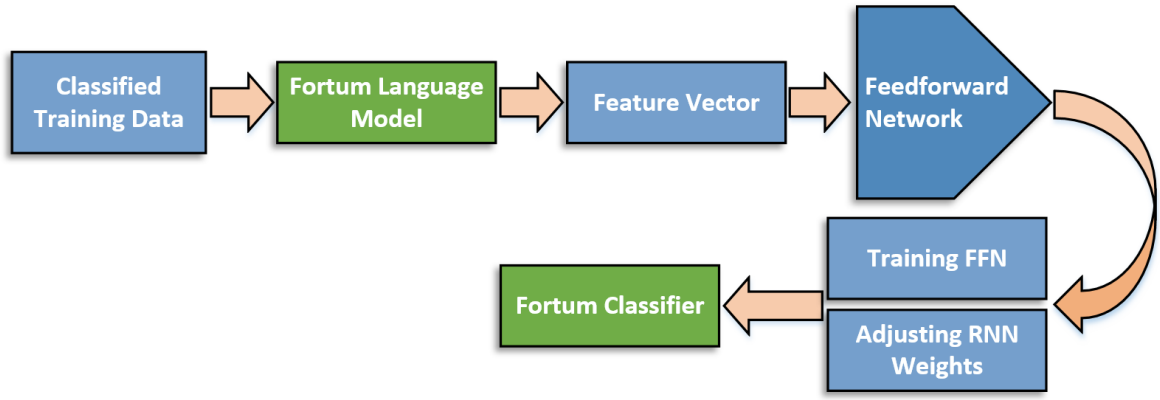


Figure 40 Workflow for training the classifier

While the input layer of the FNN equals to the output layer of the language model, the output layer of the FFN is parallel to the probability of each class as each neuron results in a real value between zero and one. This value is considered a probability of a requirement belonging to a certain category. The results can be backpropagated to the beginning to minimize the error between the prediction and ground truth, while the ground truth for each requirement is known in the training phase. Backpropagation through time was utilized, resulting in a separate computation of the cost of each time step. Consequently, the weights of the all networks (e.g., the RNN with LSTMs, FNN, and networks inside each LSTM cell) are concurrently adjusted for the classification task based on the backpropagation. In practice, the only controllable element is the loss function which is binary cross-entropy in this case.

In the testing phase, only the requirements are fed to the classifier without any predetermined class. As a result, the classifier outputs the probabilities. The weights and biases are not changed anymore, but they remain the same as adjusted in the last training stage. Figure 41 summarizes this workflow. Thereafter, the model can predict texts and sentences to the predetermined classes, given that the words and word orders are similar to the ones utilized in the training phase. In other words, any text can be fed to the classifier, but only similar texts to the training data can properly be classified. It should be noticed that the classifier consists of both the language model and feedforward network as illustrated in Figure 38. An overall and illustrative workflow for testing the algorithm is presented in Appendix 1.

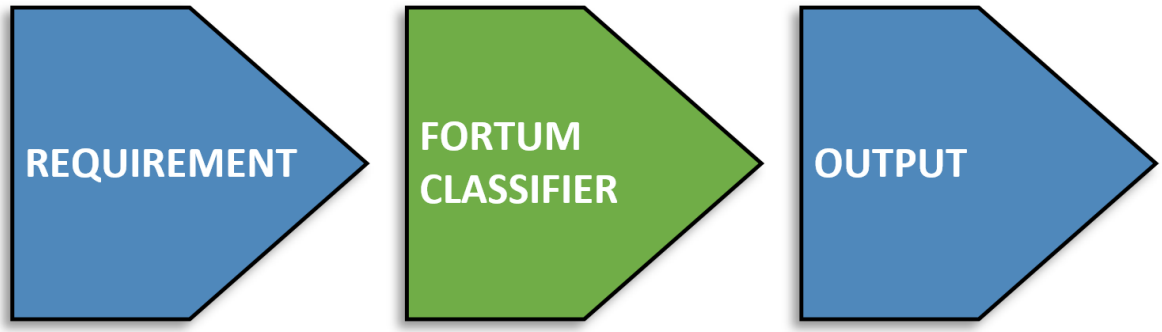


Figure 41 The general workflow in both testing and utilization phases of the classifier

As earlier discussed, the initial classified data were split into training and test datasets. The preliminary test dataset consisting of 733 randomly sorted requirements was not used when the model was trained. Otherwise if the data would have been seen beforehand, the model would be very accurate. Thus, the model was tested without showing the features and labels of the test dataset to the model in advance. Since the test set was randomly split from the original set, the requirements were similar to those utilized in the training phase.

The predefined requirements hierarchy specified the structure of the algorithm. Two different requirements levels were desired to be clearly separated. Initially, the structure was as follows: all nine classes were at the same level, and single-level classification was forced to result in two-level categorization. It was structurally considered the easiest way to constitute the arrangement. In other words, if a subcategory is selected, the requirement has to belong to the process class. However, this architecture was considered impractical.

The actual and final arrangement of the algorithm consists of two classifiers. The two-level classifier filters results belonging to the process class, and thereafter, it further classifies the process requirements into suitable subcategories. This means that a text is initially explicated and the probabilities are provided for the high-level categories. During the possible second round, the applicable subclasses are determined for the requirements originally belonging to the process class. The arrangement is illustrated below in Figure 42.

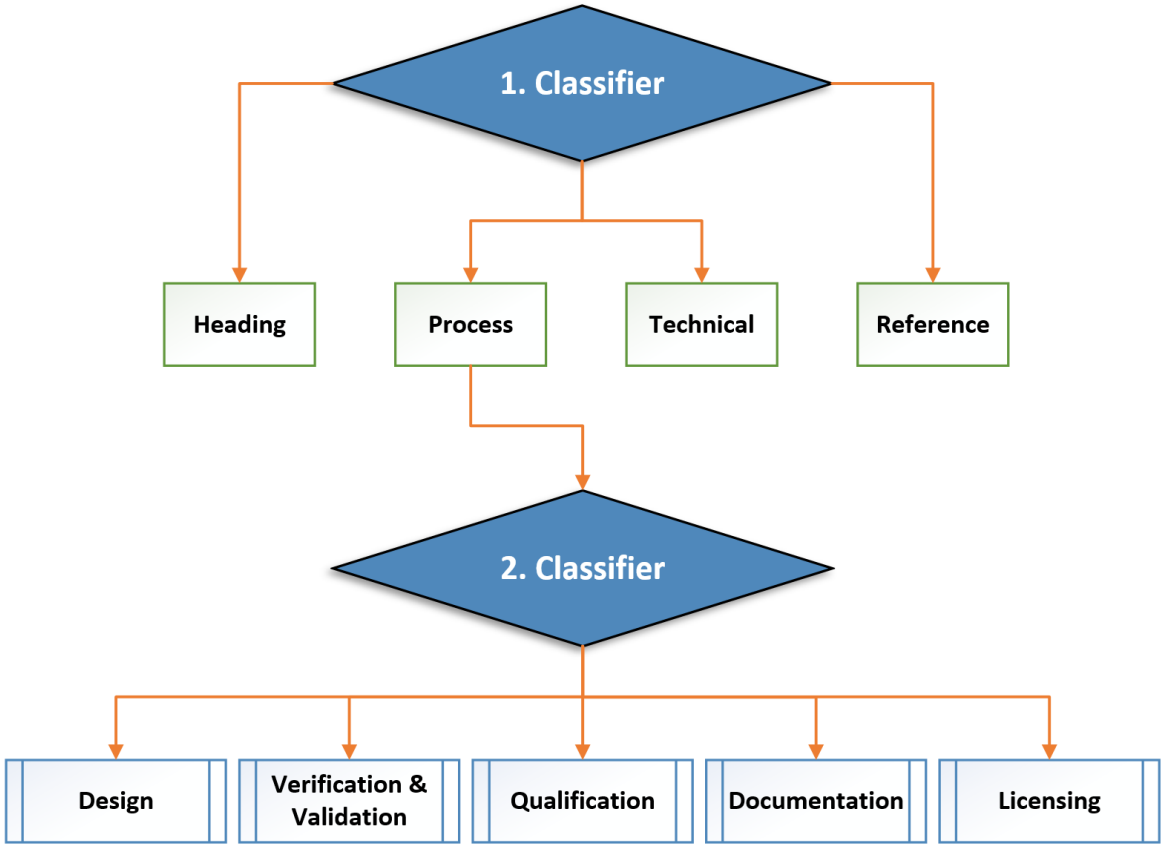


Figure 42 Hierarchy of the two-level classifier

Both classifiers include a threshold which prevents the classifier from predicting an incorrect label or labels based on the probabilities. If the probability given by the classifier for a certain class is below the threshold, no class is given. The activity is performed for every requirement according to the same procedure. Additionally, it is highlighted that “class” and “label” can be considered synonyms since they are used to represent both human and algorithm statements, respectively.

To summarize, the language model has been created utilizing transfer learning and unstructured data to train the domain language, that is, the language of the YVL Guides. Furthermore, the manually labeled examples are used to train the classifiers. Specifically, the RNN is provided with the requirements and the network outputs a representation vector. The representation vector is used as an input for the feedforward network. The feedforward network is added to the trained recurrent neural network to generate label predictions. The hidden layers include weights and biases which are adjusted based on training (i.e., backpropagation). Finally, the FFN outputs a 9-dimensionsal vector, that is, the probabilities for each class. The probabilities of the model in question are discussed more in the following chapter.

4.4 Algorithm Classification

When the algorithm had been trained with two-thirds of the initial dataset, it was tested by using several test sets, such as one-third of the initial dataset. As already discussed, the test datasets were not used in training. Thus, the requirements and the corresponding categories were totally unfamiliar to the algorithm. In total, four different test cases listed in Table 19 were arranged. Even though the test sets consisted of both the requirements and the related classes, only the requirements were entered to the algorithm. It is emphasized that the correct answers are considered ground truths, which indicate the data science perspective. The ground truths, determined by the human experts and considered as correct answers, were utilized to evaluate the accuracies of the algorithm. It should be noted that even the ground truths have been labeled by the experts to the best of their ability, the labels are dependent on the determined hierarchy, the content of each class and human interpretations. Therefore, the ground truths may not be considered as absolute answers.

The classes labeled by the algorithm depend on the thresholds which were separately iterated for each case. Due to simplicity, the thresholds were manually iterated by investigating three different values in each test case, specifically 0.5, 0.7 and 0.9. Thereafter, the most optimum value resulting in the highest score was chosen based on the optimization of the overall accuracy. The accuracies are discussed more in Chapter 4.5 Model Accuracy.

The thresholds define the level of certainty the algorithm should have to predict a label. Thus, in case a threshold is set to 0.60, the probability of a requirement belonging to a specific class has to be equal or more in order to be labeled. The thresholds determined and used in each test case are listed in Table 19. In the table, the first test case is divided into two parts because there were initially two test stages on the classified data. The first classifier was separately tested, and the second classifier was introduced afterwards. As illustrated in Figure 42, the hierarchy of the classifiers results in a separate categorization for the high- and sub-level categories.

Table 19 Thresholds used in each test case

Test Case	Threshold
1/3 of Initial Data (only high-level classes)	0.90
1/3 of Initial Data (including subcategories)	0.90
Blind Test YVL B.2 (including subcategories)	0.70
Blind Test UK SAP (including subcategories)	0.50

The objective for the execution time was set to 80 requirements in a minute, as mentioned in Chapter 1.2. During the four tests, the algorithm categorized 100 requirements in a minute. However, minor design modifications were performed afterwards, and currently, it takes one minute to classify 1000 requirements.

Next, the results of the test cases are discussed and examples of the labeling are presented. Three different cases are included, namely correct labels, wrong labels and labels which may be considered more accurate results compared to the human decisions. The latter two cases include the predicted labels which can be considered incorrect when compared to the corresponding ground truth(s). However, after a proper analysis of the results, the last case mentioned may be considered a better outcome because the algorithm has been more consistent in classifying the requirements in contrast to a human.

Examples of the correct results from the preliminary test including the subcategories, that is, the second classifier, are presented in Table 20. The table lists probabilities of the classifier in predicting each paragraph. In addition, only the classes of which given probability differs from zero are shown. These requirements represent a portion of the initial test dataset consisting of 633 requirements. Based on the probabilities and thresholds, the algorithm has anticipated labels which are shown in Table 21.

Through a configuration item (CI) ID, the labels of each requirement are linked to the requirements and their probabilities. This linkage is essential to ensure unambiguous processing of the results because the number at the beginning of each requirement text alone is not sufficient. The CI ID is not only used in this study, but also in practical requirements management providing an individual linkage for each demand, thus, facilitating configuration management.

Table 20 Examples of the correct classifications (STUK, 2013d, 2013h, 2013g, 2013a, 2013e)

CI ID	**Requirement**	**Probabilities**
YVL-B.5-6.4	6.4 System modifications in an operating nuclear power plant	Heading = 1.00
YVL-E.8-2-202	202. The system design requirements on which valve design is based are presented in the B series YVL Guides.	Reference = 1.00
YVL-E.7-3.4.2-351	351. In connection with the final suitability analysis of electrical or I&C equipment in safety class 2, an independent assessment of the acceptability of the qualification procedure shall be presented.	Process = 1.00 Qualification = 1.00 Documentation = 1.00
YVL-B.1-5.4.2-5425	5425. The plant unit's power supply systems shall be dimensioned to supply sufficient electrical power for the implementation of the safety functions in all plant conditions.	Technical = 1.00
YVL-E.7-5.3-521	521. The schedule for the delivery and installation of an electrical and I&C system, component or cable in safety class 2 or 3 shall be planned in a manner that allows for implementing the modification planning and modifications that may be required after the factory tests in accordance with procedures that are in line with the safety significance of the system or component.	Process = 1.00 Design = 0.93 Documentation = 0.03
YVL-E.7-6.7-648	648. The testing of the software shall include static and dynamic tests.	Process = 1.00 V&V = 1.00 Documentation = 0.03
YVL-E.3-5.3-513	513. The specifications shall be submitted to STUK for approval prior to the submission of the construction plans.	Process = 1.00 Documentation = 1.00 Licensing = 1.00

Table 21 Examples of the correctly predicted labels compared to the ground truths

Configuration Item ID	Predicted Label	Ground Truth
YVL-B.5-6.4	Heading	Heading
YVL-E.8-2-202	Reference	Reference
YVL-E.7-3.4.2-351	P Qf Do	P Qf Do
YVL-B.1-5.4.2-5425	T	T
YVL-E.7-5.3-521	P D	P D
YVL-E.7-6.7-648	P V&V	P V&V
YVL-E.3-5.3-513	P Do Li	P Do Li

Examples of the incorrect classifications are presented in Table 22 and the corresponding labels in Table 23. They show that the algorithm may predict a requirement class either completely or partly incorrectly. Thus, the predicted label does not match with the ground truth. The current purpose is only to present the results, whereas the accuracies of the algorithm are discussed in more detail in Chapter 4.5 Model Accuracy. Thereafter, the results are further analyzed in Chapter 5.

As can be seen in Table 23, it may be difficult to label some requirements. Especially, the second paragraph, YVL-E.6-10.3, is actually heading, also according to the STUK VAHA-A categorization (STUK, 2018c). Because of the long text and similarity to a full sentence, the algorithm may have considered it a process requirement. However, the algorithm has not been able to find any similarity with the subcategories. The first paragraph, YVL-E.3-12.4, is similar to the second one but may be more complicated to be classified due to an intractable structure of the title. The algorithm has correctly recognized that the text includes words and word order typical of both heading and technical requirements.

Table 22 Examples of the incorrect classifications (STUK, 2013e, 2013f, 2013a, 2013g)

CI ID	Requirement	Probabilities
YVL-E.3-12.4	12.4 Periodic inspections of pressure equipment subject to registration	Heading = 0.02 Technical = 0.01
YVL-E.6-10.3	10.3 Documents to be submitted at the construction licence phase	Process = 0.93
YVL-B.1-5.3.2-5313	5313. A qualification plan shall be provided for the main control room and the necessary monitoring and control posts when the application for a construction licence is filed.	Process = 1.00 Qualification = 0.73 Documentation = 0.99 Licensing = 0.01
YVL-E.7-5.6-552	552. A test of electrical and I&C equipment and cable simulating an accident shall cover exposure to radiation and stresses caused by temperature, pressure and humidity equivalent to accident conditions as well as rapid changes in the conditions.	Technical = 1.00

Table 23 Labels of the incorrectly predicted requirements compared to the ground truths

Configuration Item ID	Predicted Label	Ground Truth
YVL-E.3-12.4	(None)	Heading
YVL-E.6-10.3	P	Heading
YVL-B.1-5.3.2-5313	P Do	P Qf Do Li
YVL-E.7-5.6-552	T	P Qf

The most obvious classes of the third requirement, YVL-B.1-5.3.2-5313, in Table 23 have been labeled by the algorithm, namely process and documentation. On the other hand, the two most influential classes have not been classified, specifically qualification and licensing. Even though the probability for the qualification class is relatively high (0.73), it is below the determined threshold (0.90) shown in Table 19. Thus, it is not labeled. In this particular case, the licensing class may be questionable since initially, it was considered an action in which documents are supplied to the authority. However, as the training dataset was reviewed, similar classifications to the requirement in question could be discovered.

The last requirement, YVL-E.7-5.6-552, in Table 23 has incorrectly been explicated because it relates to type tests and the context, that is, the heading of the chapter to which the requirement belongs, is “5.6 Qualification to environmental conditions”. One reason for the misclassification may be various product related words, whereas the most essential words are written at the beginning of the sentence.

Examples of the classifications which may be considered more accurate results compared to human predictions are listed in Table 24 and the associated labels in Table 25. The algorithm has initially learned a certain way or logic of classifying requirements, and when the model is tested, it outputs results being literally incorrect compared to the ground truth. However, the predicted labels may be considered correct after a precise analysis. Therefore, even if a human has mislabeled a certain requirement, the algorithm has correctly labeled it because of the consistently labeled training data. Unfortunately, this means that some labels (or empty labels) have been incorrect, even though after a proper analysis they would be agreed correct without comparing to the human labels.

The classifier has considered the first requirement, YVL-E.3-6.2-623, in Table 24 a design requirement, whereas, a human has defined it as a technical requirement. The requirement particularly discusses design related matters, that is, the way of which the design should be performed. Therefore, the classifier has been more accurate. The second requirement, YVL-E.7-3.4.1-344, is upon a submission to STUK. Thus, it could also be classified as belonging to the licensing category. Similarly, there is only an indirect discussion about documentation in the last requirement, YVL-E.8-9-906. However, it does not require any specific action related to the documentation. For this reason, it could be justified not to belong to the documentation class.

Table 24 Examples of the classifications in which the algorithm has been more accurate than human experts (STUK, 2013e, 2013g, 2013h)

CI ID	Requirement	Probabilities
YVL-E.3-6.2-623	623. Hydrodynamic design shall be based on the process engineering requirements or other design requirements defined for the equipment or structure in question, so that the dimensioning, geometry and capacity of the components make the hydraulic operation of the system possible.	Process = 1.00 Design = 1.00
YVL-E.7-3.4.1-344	344. A component quality plan shall be presented in connection with the preliminary suitability analysis, if it has not been submitted to STUK together with the system level documentation (see para. 902).	Process = 1.00 Qualification = 1.00 Documentation = 1.00 Licensing = 1.00
YVL-E.8-9-906	906. Factory tests shall be conducted in accordance with approved procedures. Factory tests belonging to the construction inspection can be conducted once a review of the result documentation and an inspection of structure have been conducted, and an inspector of STUK or an authorised inspection body has verified the readiness for testing.	Process = 1.00 V&V = 1.00 Licensing = 0.93

Table 25 Labels of the requirements sampled in Table 24

Configuration Item ID	Predicted Label	Ground Truth
YVL-E.3-6.2-623	P D	T
YVL-E.7-3.4.1-344	P Qf Do Li	P Qf Do
YVL-E.8-9-906	P V&V Li	P V&V Do Li

Heretofore, the results of the preliminary test have been discussed, in which the dataset consisted of the randomly extracted requirements from the initial classified dataset. The first blind test was performed by using the whole YVL Guide B.2, which was also utilized in training the domain-specific language model. Despite the usage in training the language model, the algorithm had not seen any correct answer of the categorizations in Guide B.2, but only the requirements as written in the guide. The requirements were concurrently and separately classified by the experts. In this case, it was decided together with the data scientist to label two-thirds of the requirements (44) without seeing the results, and one-third (21) of the requirements with the help of the outcomes. This was due to an intention to test usefulness of seeing the results (i.e., the probabilities and the associated labels) while performing the classification task. Once the results were accessible, the classifications were compared to each other. However, a potential benefit of knowing the results could not be observed because of the small amount of requirements. Even though the second test dataset included much more requirements compared to the previous one, the similar test was determined not to be replicated during the second blind test with the UK data.

Next, examples of the results from both blind tests are presented. Table 26 lists examples of the interesting classifications in the first blind test while the corresponding labels are presented in Table 27.

Table 26 Examples of the interesting classifications from the first blind test (STUK, 2013c)

CI ID	Requirement	Probabilities
YVL-B.2-3.1-302	302. For management of the nuclear facility's safety functions, the facility shall be divided into structural and functional entities, i.e. systems. The systems shall be further divided into structures and components. The division shall be such that every structure and component affecting the nuclear facility's operation and safety shall belong to a system.	Technical = 1.00
YVL-B.2-3.1-303	303. The nuclear facility's systems, structures and components shall be grouped into the Safety Classes 1, 2, and 3 and Class EYT (non-nuclear safety).	Technical = 0.98
YVL-B.2-3.5-335	335. System boundaries shall be unambiguously indicated in the classification document's main diagrams for electrical systems and in the schematic diagrams of I&C systems.	Process = 0.97 Design = 0.86 V&V = 0.03 Qualification = 0.01 Documentation = 0.91 Licensing = 0.02

Table 27 Labels of the highlighted classifications of the first blind test

Configuration Item ID	Predicted Label	Ground Truth
YVL-B.2-3.1-302	T	T
YVL-B.2-3.1-303	T	P D
YVL-B.2-3.5-335	P D Do	P Do

The first requirement, YVL-B.2-3.1-302, is clearly interpreted belonging to a technical class because it discusses functional and structural entities. The second requirement, YVL-B.2-3.1-303, represents one of the major challenges in the classification task: a requirement including a slightly new subject not previously included in the training data. In this case, the subject is the safety class. Due to the new-found theme, it was necessary to include it in one of the classes available. After a proper discussion with the relevant experts, the design class was decided to cover the theme. It is emphasized that there are requirements in the training data discussing subjects including safety class related issues, but not exactly about defining them. However, it is not axiomatic that this particular subject could not be covered with the technical class. It is only a matter of definition.

Finally, the last requirement in Table 26 is challenging to be interpreted because the requirement consists of words and subjects similar to the content of many other classes. Therefore, the classifier may have recognized possibilities to every process subcategory. More importantly, the relevance of the design class is questionable. After the consultation of an expert specializing in the safety classes, it was recognized that the requirement does not apply to design processes but only to documentation by defining the content of a classification document. System boundaries are considered during the design, but the particular requirement does not apply to design as such. However, the raw data included many design requirements discussing systems, while particular license application related requirements were also associated with the design class. Therefore, the classifier may have incorrectly categorized the requirement.

The second test was organized and the results analyzed after the first blind test. The content and structure of the second blind dataset was described in Figure 26 in Chapter 3.1 Data Collection. Similar to other datasets, the UK requirements were analyzed in Excel and verified by the relevant experts prior to knowing the predictions of the classifier. Once the manual classifications were available, the results were compared to each other. Table 28 lists three examples of the UK blind test predicted by the model, whereas, the predicted labels and the respective ground truths are tabulated in Table 29.

Table 28 Examples of the classifications in the UK blind test (ONR, 2014)

CI ID	**Requirement**	**Probabilities**
ECE.15	Where analyses have been carried out on civil structures to derive static and dynamic structural loadings for the design, the methods used should be adequately validated and the data verified.	Heading = 0.03 Process = 0.97 Technical = 0.02 Design = 0.40 V&V = 0.58 Qualification = 0.08 Documentation = 0.39
EKP.4	The safety function(s) to be delivered within the facility should be identified by a structured analysis.	Heading = 0.06 Process = 0.94 Technical = 0.04 Design = 0.26 V&V = 0.03 Qualification = 0.06 Documentation = 0.04
EPS.4	Overpressure protection should be consistent with any pressure-temperature limits of operation.	Heading = 0.11 Reference = 0.02 Process = 0.15 Technical = 0.56
ESS.16	Where practicable, following a safety system action, maintaining a stable, safe state should not depend on an external source of energy.	Heading = 0.05 Process = 0.02 Technical = 0.80

Table 29 Labels of the UK blind test examples

Configuration Item ID	Predicted Label	Ground Truth
ECE.15	P V&V	P V&V
EKP.4	P	P V&V
EPS.4	T	T
ESS.16	T	T

It is noted that even many classes have probabilities greater than zero, the threshold was set to 0.5 as listed in Table 19. The first requirement, ECE.15, demands validation of the analysis methods and verification of the used data. Therefore, it is obvious that the requirement belongs to the V&V category. There are relatively high probabilities for the design and documentation classes which cause the categorization to be very interesting. These two categories being below, but close to the threshold, may be affected by words, such as "design" and "data". However, they are not considered relevant labels in this case.

Similarly, the model has provided probabilities for many classes in the second example, EKP.4. In the paragraph, the object is to demand an analysis, but only a minimal probability has been provided for the V&V class. The ground truth involves the same labels as in the previous example, whereas, only the process class has correctly been predicted.

The third example in Table 28, EPS.4, has the technical aspect. The requirement related to overpressure protection involves functional characteristics due to which it has correctly been categorized to the technical class. The model may slightly have considered the reference class because of the phrase "consistent with", whereas, the process category may have achieved the probability of 0.15 due to the word "operation" or even the phrase "limits of operation".

The last categorized requirement, ESS.16, of the examples is an excellent example of the well-performed classification. The algorithm has even been 0.80 certain that the requirement belongs to the technical category. Even though there have been samples involving similar matters in the training data, this classification can be considered a success. The model is able to recognize evident technical requirements even issued and formed by another authority. There are several other requirements in the results comparable to the classification of this requirement.

Altogether, given that the utilized language model was based on the YVL Guides issued by the Finnish authority, the UK requirements were classified more effectively than it was initially expected. Even though this research used the YVL requirements written in English, many differences have been discovered between the domains. Generally, the ONR dictates that despite the specific plant design, the only essential actions involve demonstrations that the design is safe. In contrast, STUK takes into account many functional and non-functional principles the plant supplier or a design organization should consider when designing a facility.

4.5 Model Accuracy

Model accuracy indicates the performance of the algorithm, that is, the higher the accuracy, the better the performance of the model. The performance of the model may be analyzed from various perspectives leading to different scores depending on the evaluation metrics. In this research, Hamming loss function, widely used in multi-label cases, was utilized to measure the capability of the algorithm. Additionally, other possibilities to calculate accuracies were discussed, namely precision, recall and f1-score, the latter one being a combination of the first two. It was important to discuss about the most relevant way to indicate the practicability of the results because each metric emphasizes different aspects.

After each test, the model was scored by comparing the predicted labels against the ground truth(s). In the preliminary tests, the ground truths were included in the datasets as the test dataset was formed by dividing the initial classified dataset into two collection. Conversely, in the case of the blind tests, the correct labels were determined separately from the prediction results. This chapter discusses the overall accuracies of the model separately determined for each test case.

The test cases with the corresponding thresholds as well as the reported fraction of the wrong labels to the total number of labels, that is, Hamming loss, are tabulated in Table 30. In addition, based on the Hamming loss, the corresponding accuracies have simply been calculated as (1 - Hamming loss). The results have been rounded to the nearest hundredth, and each fraction also converted into percent. It is emphasized that the determined threshold only affects to the predicted labels, not probabilities. Thus, the probabilities of each class remain the same, and only the presented labels depend on the threshold. However, only the labels matter from the accuracy perspective.

Table 30 Model accuracy and the corresponding threshold in each test case

Test Dataset	Threshold	Hamming Loss	Accuracy
1/3 of Initial Data (only high-level classes)	0.90	0.10	0.90 (90 %)
1/3 of Initial Data (incl. subcategories)	0.90	0.20	0.80 (80 %)
Blind YVL B.2 (incl. subcategories)	0.70	0.14	0.86 (86 %)
Blind UK SAP (incl. subcategories)	0.50	0.15	0.85 (85 %)

In each case, three different thresholds, namely 0.5, 0.7 and 0.9, were manually tested and the corresponding hamming loss evaluated. Based on these iterations and the results, a threshold resulting in the smallest error (i.e. the highest accuracy) was chosen. The manual evaluation was considered sufficient to properly represent the efficiency of the model at this stage. When investigating the behavior of the model by varying the threshold, only the UK dataset retained the same performance between the thresholds 0.5 and 0.7.

The ROC curves were separately determined for each class to further evaluate the classifier in the case of the first blind test with the YVL Guide B.2. Even though the ROC curves including all thresholds, and briefly discussed in Chapter 2.3.1, were plotted for every label, only two examples are illustrated as follows. To compute the simple curves, the multi-class problem was transformed into a binary classification. Consequently, the performance was individually explicated in all cases.

Figure 43 represents the first ROC curve which indicates that the process classes have been comparatively well detected from all classes, whereas, Figure 44 indicates that there have been difficulties with labeling technical requirements in the blind set. The proportion of the correctly classified technical requirements is almost equal to the proportion of the incorrectly classified samples that are not technical. This can actually be seen from the results that the algorithm becomes confused with the process requirements. Given that the subjects of the requirements in the Guide B.2 are mostly dissimilar compared to the requirements in the training set, the confusion is understandable.

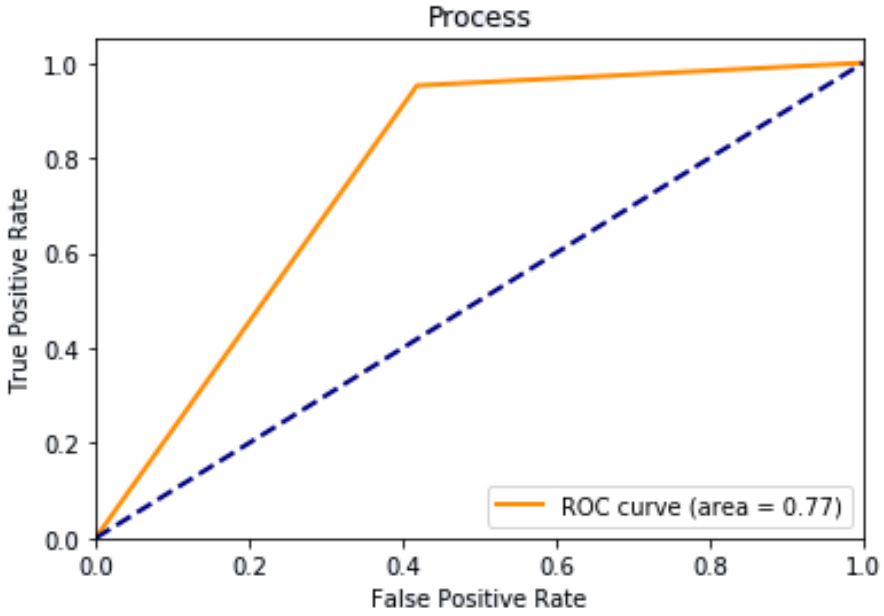


Figure 43 Receiver Operating Characteristics and the area under the ROC curve for the process class in the first blind test

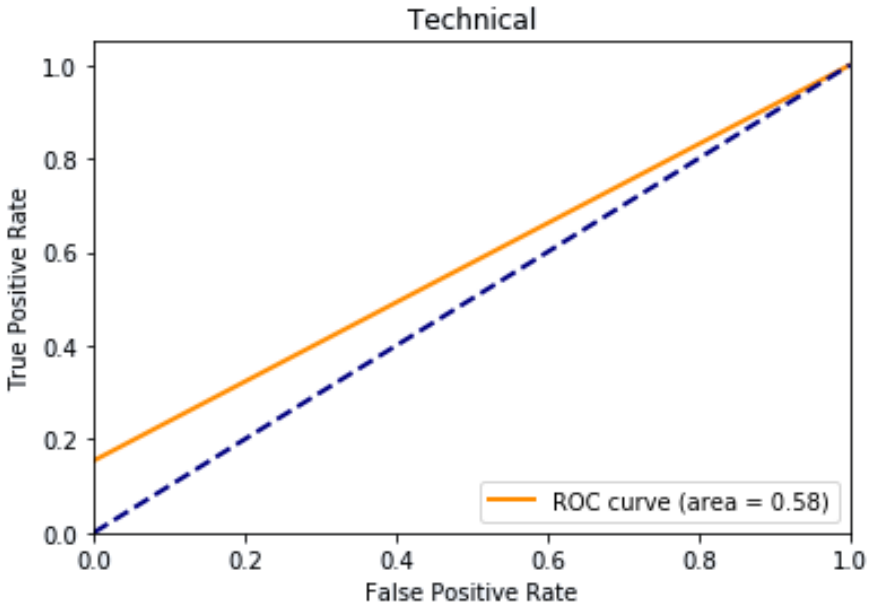


Figure 44 Receiver Operating Characteristics and the area under the ROC curve for the technical class in the second blind test

5 Discussion

This chapter discusses the main challenges of the study, future research possibilities, and limitations of the study. In the case study, weak artificial intelligence was utilized focusing on a narrow task. In practice, the task was to process natural language and classify text according to the examples used in training the model. This was a narrowly defined specific problem which should be considered when analyzing the future possibilities in this particular field, which is not, however, restricted to nuclear power specific language but rather natural language in general.

Already at the beginning of the research, the objective of the execution time of the algorithm to classify requirements was exceeded by 25 percent. Eventually, through minor design modifications, the execution time reached 1000 requirements in a minute, that is, the percentage increase of 1150 %. Thus, the current level of the computational performance is suitable for focusing on more influential actions related to requirements analysis.

In addition to the high computational performance, the model exceeded the accuracy level of 0.7, which was initially determined as an objective for each test case. The achieved accuracies are significant considering the amount of training data and the nature of the requirements. However, many improvements are required to better facilitate the usage of the algorithm. Next, both existing challenges and future possibilities to further develop the results of this thesis are discussed.

5.1 Challenges

Even though the research resulted in many positive outcomes, there are challenges associated with this study. First of all, the current parameters (i.e., weights and biases) cannot be employed in any other case, such as classifying requirements into other categories than those utilized in the training dataset. This also illustrates the weakness of narrow artificial intelligence.

Due to the simplification of the case study, the utilized requirement categories are not practicable in the more detailed classifications, mostly because the technical requirements, which have not been further classified in this study, form the basis of the ADLAS methodology. For this reason, there is no need to classify all the YVL Guides according to the developed hierarchy. This applies both to the manual and automatic categorization. However, the most of the YVL Guides together with other documents, such as standards, could be employed to classify the requirements according to the ADLAS hierarchy. Thereafter, the algorithm could be utilized, for instance, in classifying the requirements of other licensing domain or reclassifying the YVL requirements in case of a revision.

A major challenge involved in the categorization of the requirements was dissimilarity to any other requirement in the training dataset. Thus, the test datasets which involved unfamiliar terms compared to the ones used in the training set, performed at a lower-level. There were many requirements in the test sets that the model did not identify because they consisted of new words and word orders. Headings are the simplest example of this challenge. Occasionally, they may be unusually individual: there may not be any similar texts corresponding to the heading class in the training data. Thus, the model has not learned to recognize such paragraphs. Furthermore, safety and seismic classifications in the test set

of the YVL Guide B.2, and safety classes in the UK SAP dataset emphasize this challenge.

In the software industry, it has been noted that in case architecturally significant requirements (ASRs) are initially wrong, incomplete, inaccurate or lack details, errors may have arisen (Chen, Ali Babar, and Nuseibeh, 2013). The same challenge applies in the nuclear industry and especially in requirements engineering. Relevant requirements should be rapidly and accurately identified at the beginning of a project. Even if an AI-model was utilized, it can result in incorrect predictions, which may or may not be recognized sufficiently early by an expert. This sets challenges for the model to avoid predicting so called false negatives, which were earlier discussed theoretically in Chapter 2.3.1 Machine Learning. This is important because they will not be easily noticed in the case of the categorized requirements being filtered according to the desired class and some essential requirements lacking the related label, in other words, the predictions are false negative. False negatives must be avoided especially regarding safety critical requirements. These requirements could be preferably identified by labeling extra classes rather than avoiding labels. However, it is recognized that only relevant labels should contribute, thus accurately predicting relevant labels would be more important than predicting irrelevant ones.

In the classification task, several related requirements were categorized based on the precursor of a certain requirement. Although a certain requirement might contain no relevant words in the case of the last requirement, it was decided to classify the requirements according to this procedure. For this reason, the datasets consisted of requirements whose categorizations required knowledge of the content. Therefore, the discovery of similarities may have been challenging between specific classifications. In addition, considering the design class, that is, the subcategory of the process class, the quality requirements could potentially be a separate subcategory in the future. This is due to their amount in the current datasets as well as the identification of additional sources among the YVL Guides.

When the classes and hierarchy were determined and even during the labeling task, it was challenging to distinguish between verification and validation, and qualification since their actions are very similar to each other. For instance, the preliminary test results, when both classifiers were available, show that out of 61 qualification labels (ground truths), 22 have been labeled by the algorithm. Furthermore, there are four requirements which have been categorized as being part of the qualification class; however, three of them are clearly incorrect, and the fourth one is questionable.

The documentation class has been more supportive than self-contained; rather, it has supported other classes. There are only a few requirements related to documentation management and some portion of requirements describing the content of various documents. There are many requirements in the YVL Guides containing references to other guides, standards and even chapters in the same guide. Therefore, the reference class has also been used to support the classification task, and to test the ability of the model to discover such requirements and indicate their affiliation to this class. It seems that the classifier has been able to learn a similar logic; however, partly due to the incomplete consistency, not every reference has been recognized either by a human or the classifier.

As the YVL requirements have been descriptively written, they do not fulfill the quality requirements or characteristics defined by ISO 29148 Standard and discussed in Chapter 2.1.4. The guides contain challenging and ambiguous requirements, the analyses of which

are time-consuming. A human may struggle in trying to understand the real demands of a requirement. Table 14 listed examples of such requirements. Cases similar to ones represented in Table 14 and the corresponding analysis decrease the efficiency in a requirements classification task, thus, presenting challenges in requirements analysis. The international situation reveals national differences in safety regulations with various cultural practices impeding the harmonization of regulatory licensing regime (MIT, 2018).

When the model was tested against the UK Safety Assessment Principles (SAP) dataset comprising of 317 requirements, it was observed that the model could not properly recognize the subcategories. Although the current outcome is a good result because the model correctly labeled most of the process classes, the more essential labels, that is, the subcategories were lacking. Initially, it was known that the content of the requirements varied compared to the requirements in the YVL Guides. In the SAP, the Office for Nuclear Regulation uses distinctive terms, such as safety case, compared to the YVL terms in the training dataset. Unfortunately, STUK also uses the term "safety case" but in a guide which was not included in the data collection of this thesis. For this reason, the training dataset of YVL Guides lacked the characteristic words of SAP. Consequently, additional determinations for the content of the classes were performed when classifying the UK requirements.

Common challenges have been recognized when moving from one domain to another, that is, one distribution of words to another. They include a different vocabulary with the varying context and various words describing the same subject. However, the tested model was only trained with the YVL Guides; thus, it represented only a specific domain and its language. Additionally, most of the high-level categories were correctly detected, which is a good result, given that the requirements are variously composed. It was discussed whether the classifier should provide at least the label having the probability closest to the threshold in case there is no subcategory labeled. Nevertheless, it was decided that the current language model be fine-tuned with the UK domain language, after which the tested dataset could be reused for testing the model against the specialized language model. It is emphasized that the feedforward neural network part, including its weights and biases, remains the same.

Furthermore, an ethical perspective should be considered in the development of an AI model, which raises many questions, such as those listed below.

- What are the actions the model should perform?
- Which issues should not be performed by the algorithm?
- How can the algorithm be guided to a correct and safer direction whenever the prospects of artificial intelligence and machine learning are increasingly enhanced, and the respective technology solutions will become even more spontaneous?

In answer to the latter question, it could be suggested that whenever requirements are verified, the classifications and elicited requirements should be coherent and complete. The demand for completeness of the requirements implies that they must be precisely determined directly the requirements analysis process has been initiated. Thus, false negatives should be avoided in the requirements classification. In addition, every elicited requirement should be clear, easy to understand, complete and consistent. Having the correct requirement specification also enhances the traceability including precise relationships between the requirements at the beginning of the project.

5.2 Future Possibilities

Many future development possibilities were recognized during and at the end of this research. The possibilities relate to the utilization of natural language processing algorithms in requirements analysis. The feasible suggestions focus on supervised machine learning. The requirements analysis is generally a challenging process because of ambiguous statements. The challenge is not a country-specific issue because it involves internationally all parties, that is, the regulating authority, supplier, and licensee or client.

One of the most promising applications in which this method could be used is Fortum's Advanced Licensing and Safety Engineering Method (ADLAS), briefly described in Chapter 2.2. When combined with the ADLAS methodology, the algorithm could be utilized in advancing both internal projects and the needs of external customers. The requirements, regardless of the parent-child relationship, should be categorized to the relevant lifecycle stages. Although the definition of the lifecycle can be separately determined for each project, there is always a development, manufacture and verification stage to some extent.

Despite investigating the next steps towards the development of ADLAS, there also are several other targets needing development in the area of requirements engineering. Partly related to each other, the other tasks are first discussed, following by the proposals for the subjects closely related to the basics of requirements analysis.

Original or native requirements (e.g. YVL requirements as such) including references could be analyzed further to specify the scope of requirements. When a reference is recognized in a requirement, additional categories for the reference should be defined. The scope of requirements specification would be easier to be determined once the other references are known, that is, the YVL Guides to which the requirement refers. For instance, in the case of a specific task, it would be necessary to quickly recognize only the relevant requirements, denoting the relevant YVL Guides which should be taken into account when performing the task. Furthermore, it would be useful if the model could emphasize the next requirements or materials essential to continue the requirements analysis after the initial classification. This would be very helpful and effective if the algorithm could be asked to provide requirements as follows: "I would like to have each requirement related to a pump, please show them to me".

In case a requirement only describes something instead of requiring an action, and refers to another source, the model should be capable of recognizing that the text itself is not a requirement but it comes from somewhere else. Furthermore, when referring to graphs and figures, the algorithm could highlight that "see this particular figure" or "act according to this description".

When the model does not understand the meaning of the text, it could recognize its incapability and highlight such requirements, which an expert would easily be able to collect and classify manually. This could involve a possibility to manually alter the threshold, and provide information about its uncertainty regarding a specific class, for instance, by formatting the cell to be distinguishable from the certain ones. In addition, this subject relates closely to the challenge in which there may be unfamiliar terms in new datasets, specifically, words not included in the training set. Therefore, a keyword list could be valuable, also supporting the cases in which some categories contain sparse labels.

An AI-model could be utilized to combine similar requirements from different requirement sources and further elaborate them. In the latter case, the context should be first recognized to be capable of understanding the semantic meanings of words. Revealing the meanings is considered an easier assignment even for an expert than being able to elaborate requirements, that is, forming and writing new ones. Currently, elaborating requirements would imply that it may also be necessary to atomize them. The atomization is considered an essential task due to the ambiguous and vague requirements whose existence has also been recognized during the requirements analysis and classification assignment of this study. The examples of these demands were represented in Table 14 in Chapter 4.2.

Even though the objectives of the research were set to discover future possibilities related to natural language processing, and the research examined the supervised NLP method in requirements analysis, several other possibilities were revealed touching upon these issues. The first interesting topic is utilizing unsupervised learning to explore possible groups of requirements that are not evident for an expert, but might be useful in practice. This means that among a large set of requirements, there may be similarities and interconnections between the requirements which the experts cannot even consider. Discovering these similarities could be possible for an algorithm but rather impossible for a human. It should be emphasized that these unrevealed categories are conceivable to be formed, but distinctive from the classes of systems engineering processes as well as various product related groups. Furthermore, patterns created by the unsupervised learning method could be used to verify the categories and their content. Naturally, the ability of an unsupervised learning model to discover similarities depends on the writing style of the requirements. Furthermore, the need for the style may differ between a human and model.

In Finland, STUK sets requirements for operational limits and conditions (STUK, 2013b). The NPP licensees have to determine and form suitable requirements with which they would comply each demand in every operational phase, such as start-up and shutdown conditions. These operational limits and conditions are presented to the authority for an approval. For instance, there are requirements which systems shall be in operation and at which stage. The manager of operating personnel is always responsible for fulfillment of these requirements. It would be useful if AI was used for expressing the relevant requirements at each stage as well as informing whether the suitable condition for each requirement has been achieved or not. However, its utilization in a nuclear power plant environment is slightly unclear and has not been further clarified.

In construction sites and operating plants, various observations are made concerning, for instance, occupational health and safety, operation or maintenance. These observations are usually analyzed and categorized by only few persons. There could be a possibility to categorize the observations similarly to any other customer feedback classification. Furthermore, this would closely relate to processing non-conformities in quality management. Related to the interpretation of the reports, in case a breakage is reported, an algorithm could even analyze the places the breakage has an effect on as well as highlight the influences on the related piping and instrumentation diagram. Thus, certain valves could be closed taking into account that the event can extend to several directions in the piping. In contrast to electrical design, in which the analysis process is more straightforward. For instance, if a switchboard breaks down, the impacts can be easier traced because of the direct connections between each component. However, this would require the utilization of an image recognition algorithm, the subject not included in the scope of the thesis.

Many dissimilar classifications compared to the ground truths were discovered during each test phase as discussed in Chapter 4.4. Because the intention of this study was not to produce profound analyses of the algorithm classifications, more detailed information could be provided in later studies. Several types of faults may be recognized, thus, their natures could be analyzed to investigate possible recurrent errors: has the model trouble repeatedly classifying certain labels? In the case of a recurrent fault, understanding the reason would enable the improvements of the model. Furthermore, even the 400-dimensional vector space is recognized to be enormous, it would be interesting to compare the embeddings of similar words to observe possible evident correlation between them.

While the execution time of the algorithm is currently 1000 requirements in a minute, some improvements are essential to be performed in the long term. Parallel computation is considered one possibility to enhance the performance. The process could be advanced by adding parallel graphics processing units. The physical location and data security of the computational server have to be precisely considered regarding to confidential or secret information, such as contract requirements analyzed by the algorithm. In addition, the reliability of the service provider has to be ensured by auditing.

Above, many future possibilities were discussed concerning the utilization of artificial intelligence. However, there are many practical requirements analysis methods which could be first researched considering the achieved results. Because the initial results in classifying requirements are promising, proposals for the future research have been recognized. The interesting development tasks could include post-processing the results from the trained models, further testing of the trained models on new datasets, ways to continue improving the accuracy of the results, and additional functionalities that would allow more efficient processing of requirements. Five specific work packages listed in Table 31 are proposed for further research. Proposals for the content of these packages are discussed as follows.

Table 31 Proposals for the future research

Work Package	Subject
1	Results handling & post-processing
2	Analyzing UK SAP requirements with the domain language model
3	Testing analysis methods for combining similar requirements
4	Testing methods for atomization of complex requirements
5	Evaluating different ways to improve the results quality in use and over time

In the first work package, each label should be presented in an individual column and the requirements that are below but close to a threshold highlighted. In addition, the possibility for the trained model to miss relevant label(s) associated with a requirement should be decreased, for instance, by modifying the cost function of the trained model specifically to avoid false negatives, that is increasing model recall. The resulting updated model would be tested and compared the changes in false negatives to the ground truth.

The model was decided to be further developed once it was noticed that the algorithm could relatively well classify the UK requirements with the YVL language model. Thus, the suggestion is to retrain the language model with the UK licensing domain documents, such as the ONR Technical Assessment Guides and Safety Assessment Principles, in the second

work package. Consequently, the UK language model is expected to result in better predictions. Furthermore, the utilization of a bidirectional language model is proposed to be investigated for enabling the usage of left and right contexts. Thereafter, the UK SAP requirements could be fed into the new language model probably resulting in slightly different feature vectors compared to the current model. Since the classifier initially recognized higher-level categories in the last testing part, the subcategories are expected to be better explored with the help of the domain-specific language model.

Based on two sets of similar requirements, the potential of automatically combining similar requirements is analyzed and evaluated as a part of the third package. The similar requirements could be combined as candidates for requirements to be elaborated into one. The work package consists of several tasks. Initially, data are explored and prepared, following by implementation, training and testing similarity analysis of two separate methods, namely recurrent neural network based word embeddings and transformer network based word embeddings. The latter one could increase the contextual and syntactic capabilities for similarity analysis. Combining the requirements as well as atomizing them may also involve elaboration of new demands. Therefore, both tasks should consider the EARS when expressing requirements.

Next work package, namely the fourth package, would test the methods for atomization of complex requirements. This could utilize the trained model to process a single YVL Guide for test purposes to atomize the complex requirements, such as requirements consisting of lists or several sentences, into separately analyzable entities, that is, full sentences. Furthermore, every multi-label classification, especially ones associated with both technical and process categories, is worthwhile to be atomized. That is due to ambiguous requirements demanding special tasks and also involving other functions even without directly stating that. Thereafter, the classified model could be run on the atomized YVL data, and test the ability of the model to detect the same and correct classes.

The last work package proposed is about evaluating ways to improve the results quality. The goal is to evaluate the different ways to increase the quality of the results using both active learning as well as data programming and weak supervision. The first method means that the model would learn better in use, while the latter one is about enriching the training data with natural language explanations. The explanations have been stated to be worth of 100 labels each, thus, decreasing the amount of labeling work required for the proper training of the model (Hancock *et al.*, 2018). In practice, re-training the model using the validated requirement labels represents active learning, whereas, data programming and weak supervision include the evaluation of the gathered natural language explanations, and for the applicable ones, re-training the model by utilizing them.

The unused testing data could be utilized for re-training which would enable better performance by improving the accuracies of the classifiers. This is considered a safe choice because the algorithm has already been tested against the data, and the results are known. The purpose of testing the model with two separate datasets was based on emphasizing the trustworthiness of the research. The applied testing procedure is represented in Appendix 1 and the workflow discussed in Chapters 3.2 and 4.3. Generally, it is known that the more classified and high-quality data, the better performance of the algorithm.

5.3 Limitations of Research

This research covered only the development of a weak artificial intelligence algorithm utilizing natural language processing to classify nuclear power industry specific requirements into certain predetermined categories. Deep supervised learning method together with the domain-specified language model were used to train the classification algorithm. Specifically, the natural language processing algorithm utilized only the long short-term memory network and feedforward neural network. The analyzed and tested requirements were considered native requirements, also referred to as entries. The task was considered a multi-label case, in which a requirement can be concurrently associated with several categories.

In the training phase, the requirement sets included YVL Guides issued by the Finnish Radiation and Nuclear Safety Authority (STUK). The specified language model was created utilizing the Wikipedia language model and transfer learning to obtain the model understanding the domain language. The datasets used for both language models were tabulated in Table 9. Finally, two blind tests were arranged to evaluate the practical ability of the classifier. The datasets used in the blind tests were listed in Table 10. In the second blind test, the requirement set included the Safety Assessment Principles (SAP) issued by the Office for Nuclear Regulation in the United Kingdom. It is emphasized that only the main principles from the UK dataset were collected and analyzed, as described in Figure 26.

The determined hierarchy for the classification included four high-level and five sub-level categories. The sub-level classes were only associated with the process category. Thus, in the case of a requirement being associated with the process class, at least one subcategory has to be attributed. Even in a multi-label classification, a requirement may concurrently belong to both technical and process categories.

As mentioned already at the beginning of the thesis, the research utilized only the deep supervised learning method. Consequently, the unsupervised learning method was kept outside of the scope of the research. However, one future possibility concerning the unsupervised learning was recognized and only briefly discussed in the previous chapter.

6 Conclusions

The objective of this research was to create a requirements categorization algorithm by utilizing the deep supervised learning method. Furthermore, the aim was to improve and maintain expertise by increasing the understanding of systems engineering processes and artificial intelligence. In the future, the natural language processing algorithm utilized in requirements analysis could decrease time and costs in the long-lasting and complex nuclear power plant projects.

6.1 Research Summary

Requirements analysis is generally initialized by defining the scope of requirements specification and analyzing as well as categorizing requirements in a pre-determined manner. The knowledge of artificial intelligence and systems engineering was developed in the manner of a literature review. The development of the algorithm was performed in collaboration with the startup company called Selko Technologies Oy, which focuses on applying artificial intelligence in requirements engineering. Generally, the thesis worker was responsible for the overall project supported by the internal steering group. However, the main responsibility of Fortum was to provide both the training and testing datasets, whereas Selko was in charge of developing, training, and testing the algorithm.

This research discovered many findings, such as a requirements hierarchy according to which several sets of requirements were classified both manually and automatically, as well as a classification algorithm and its accuracies. The study was initialized by defining the requirements categories and hierarchy followed by data collection and classification of the requirements. Once collected, categorized and verified, the classified dataset was supplied to Selko.

The developed natural language processing algorithm categorized nuclear power industry specific requirements, specifically the YVL Guides issued by the Finnish Radiation and Nuclear Safety Authority (STUK). The model was trained with the classified requirements, then tested with separate requirements both involving the predetermined hierarchy. While the results of the first tests implied valuable abilities of the algorithm, the model was also tested on another set of requirements from the UK authority, each utilized set being written in English. The aim of the second blind test was to analyze the level at which the model recognizes requirements in a different domain. All accuracies are remarkable especially when compared to the consistency of the judgments of other classification studies.

The results indicate that the current technology of deep supervised learning and natural language processing are sufficient to be utilized in the requirements classification tasks. Generally, the concern is the quality of the data and the amount of classified requirements available for training. Furthermore, the ability of recognizing various levels of hierarchy appears to be comparatively adequate. In addition, it is emphasized that the more classified and high-quality data are used for training, the better performance of the model.

6.2 Practical Implications

The research work was initialized by clearly defining the task. At the beginning, the only implemented procedure was the Wikipedia language model, and other elements of the model were customized to perform the defined task. The algorithm is only capable of recognizing nuclear power industry-specific requirements and classifying them according to the pre-defined categories; that is, it performs impressively well on carefully selected examples. The accuracy of the classification task decreases when analyzing requirement sets on which the model has not been trained. Moreover, the model is incapable of performing any other problem than categorization tasks on a specific domain. Therefore, this study solved merely the effective allocation of the requirements, while engendering many other targets to be researched in the future.

The evaluation of the current AI application is based on the questions mentioned at the end of Chapter 2.3. In reference to those questions, the outcome of the research is considered successful; the application solves a real problem and establishes new opportunities. The structure of the classification model will be retained regardless of the case study. Eventually, only the adjusted weights and biases, that can be exported to a separate file, define the way requirements are classified. Thus, the success of the categorization task is mostly due to the classified dataset and its consistency, given that there is a sufficient deep learning algorithm utilizing natural language processing in place.

As mentioned, the application already solves a real problem as it can classify requirements into specific categories at the auspicious accuracy levels established in Chapter 4.5 Model Accuracy. The long short-term memory cells have been indicated to be capable of recognizing long-term temporal dependencies and compositional semantics. However, it is emphasized that the amount of categories and data have been relatively small. More high-quality data and precise classes are required to better facilitate the needs of systems engineering in nuclear industry. Furthermore, a more complicated hierarchy structure, including several subcategories, are needed to enhance the practical requirements analysis.

The better utilization of artificial intelligence in requirements analysis involves further research. For this reason, five practical development phases in the form of work packages were recognized and listed in Chapter 5.2. Table 31 tabulates these proposed packages following by the descriptions of each content. The following studies are essential in developing a model capable of performing according to the ADLAS methodology. Therefore, classifying requirement entries is not alone sufficient, but the requirements should also be further elaborated. In practice, this means that the model would elicit and classify a requirement entry as well as elaborate it into the lowest hierarchical level, that is, the component level, perhaps through combining and atomizing certain requirements.

The NLP algorithm developed during the case study and the involved discussions revealed many future development opportunities both in nuclear engineering and other fields, including hydro and legal. There are wide intentions to further develop natural language processing algorithms to meet modern challenges of utilizing cost-effective and adaptable AI models. Natural language processing algorithms could also be developed for other requirements analyses of safety critical systems. Fortum's safety engineering knowledge combined with artificial intelligence will increase the internal efficiency and enable the development of new customer products.

Similar projects require expertise, such as compilation and processing of the crucial data. The long-lasting projects include many designing tasks and may involve several parties. Similar to any other project, the same courses of action apply to AI activities advancing the performance.

The specified research questions set at the beginning of the study were answered in the empirical part of this Master's thesis. In addition, the data and the performance of the algorithm were analyzed and discussed to perceive the current capability of artificial intelligence and natural language processing methods in requirements analysis. The findings for each research question are summarized below in Table 32.

Table 32 Summary of the main findings

Research Question	Answer
RQ1: What is the current stage of utilization of AI?	→ AI and NLP can already be utilized in requirements analysis. The current methods are capable of explicating natural language, thus, being able to categorize texts into predetermined classes.
RQ2: Where could AI and NLP be utilized along the life cycle?	→ Parts in which AI and NLP could be utilized include requirements analysis (e.g., categorization, atomization and combining), checking the fulfillment of requirements and classification of observations.
RQ3: What should be developed to better facilitate the utilization of AI?	→ More high-quality classified data (even from various sources) and secured computation capacity are required to better facilitate the utilization of AI. Furthermore, methods to combine similar requirements, atomize complex ones, and provide the results in a useful format should be researched.

The utilization of artificial intelligence is a widely discussed topic in many industries as the technology develops rapidly. This study has provided valuable knowledge about the possibilities to further develop and deepen the understanding on the systems engineering processes and artificial intelligence methods. As the results of this study are promising, the research is recommended to be continued. There are several practical proposals for the future studies which would facilitate the requirements analysis process in nuclear power industry, thus, providing a vast amount of savings in the costs and time used in the complex licensing and engineering processes.

References

Al-Rifaie, M. M. and Bishop, M. (2012) 'Weak vs. Strong Computational Creativity', *Proceedings of the AISB 2012: Computing and Philosophy*. Available at: https://www.researchgate.net/publication/262223177_Weak_vs_Strong_Computational_Creativity.

Alanen, J. and Salminen, K. (2016) 'Systems Engineering Management Plan template V1', *Research report VTT*. Available at: https://www.vtt.fi/inf/julkaisut/muut/2016/VTT-R-00153-16.pdf.

American National Standards Institute (2004) *Project Management Body of Knowledge*. Project Management Institute, Inc. ISBN: 1-930699-45-X .

Anusuya, M. A. and Katti, S. K. (2010) 'Superficial Analogies and Differences between the Human Brain and the Computer', *IJCSNS International Journal of Computer Science and Network Security*, 10(7), pp. 196–201.

Arpit, D. *et al.* (2017) 'A Closer Look at Memorization in Deep Networks', *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*. ISBN: 1873-7897 (Electronic)r0887-6185 (Linking) ISSN: 1938-7228 doi: 10.1016/j.janxdis.2011.04.006.

Baker, R. D. (1991) *Time-Cost Relationships in Construction*. Gainesville, Florida. Available at: https://apps.dtic.mil/dtic/tr/fulltext/u2/a240757.pdf.

Bar-Yam, Y. (1997) *Dynamics of Complex Systems*. 1st edn. Addison-Wesley. ISBN: 0-201-55748-7 .

Bengio, Y. *et al.* (2003) 'A Neural Probabilistic Language Model', *Journal of Machine Learning Research*, 3, pp. 1137–1155. Available at: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf.

Bingham, W. V. D. (1937) *Aptitudes and Aptitude Testing*. New York: Harper & Brothers.

Bishop, C. M. (2006) *Pattern Recognition and Machine Learning*. 1st edn. Edited by M. Jordan, J. Kleinberg, and B. Schölkopf. New York: Springer. ISBN: 9780387310732 ISSN: 15206904 doi: 10.1021/jo01026a014.

Boehm, B. W. (1981) 'Software Engineering Economics', *IEEE Transactions on Software Engineering*, 10(1), pp. 4–21. doi: 10.1007/978-3-642-48354-7_5.

Boehm, B. W. (1988) 'A Spiral Model of Software Development and Enhancement'. TRW Defense Systems Group, pp. 61–72. ISBN: 0018-9162 ISSN: 0018-9162 doi: 10.1109/2.59.

Boldon, L. M. and Sabharwall, P. (2014) 'Small Modular Reactor : First-of-a-Kind (FOAK) and Nth-of-a-Kind (NOAK) Economic Analysis Idaho National Laboratory Summer 2014 Report', (August).

Bradbury, N. A. (2016) ‘Attention span during lectures: 8 seconds, 10 minutes, or more?’, *Advances in Physiology Education*, 40(4), pp. 509–513. ISBN: 1043-4046, 1522-1229 ISSN: 1043-4046 doi: 10.1152/advan.00109.2016.

Bradley, A. P. (1997) ‘The Use of the Area under the ROC Curve in the Evaluation of Machine Learning Algorithms’, *Pattern Recognition*, 30(7), pp. 1145–1159. Available at: https://www.cse.ust.hk/nevinZhangGroup/readings/yi/Bradley_PR97.pdf.

Brank, J. *et al.* (2002) ‘Interaction of Feature Selection Methods and Linear Classification Models’, *International Workshop on Text Learning, in Conjunction with International Conference on Machine Learning*. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.20.6031.

Broadridge (2017) *What are the Applications for Artificial Intelligence in Securities Finance and Collateral Management*. Available at: https://www.broadridge.com/_assets/pdf/broadridge-what-are-the-applications-for-artificial-intelligence-in-securities-finance-andcollateral-management.pdf.

Bures, T. *et al.* (2012) *Requirement Specifications Using Natural Languages*. doi: D3S-TR-2012-05.

Cardoso-Cachopo, A. and Oliveira, A. L. (2003) ‘An Empirical Comparison of Text Categorization Methods’, in *String Processing and Information Retrieval. SPIRE 2003. Lecture Notes in Computer Science*. Springer-Verlag Berlin Heidelberg, pp. 183–196. ISBN: 978-3-540-39984-1 doi: https://doi.org/10.1007/978-3-540-39984-1_14.

Cauchy, A.-L. (1847) ‘Méthode générale pour la résolution des systèmes déquations simultanées’, *Compte Rendu des Seances de L’Acad’emie des Sciences*, 25(2), pp. 536–538. Available at: https://cs.uwaterloo.ca/~y328yu/classics/cauchy-en.pdf.

Chen, L., Ali Babar, M. and Nuseibeh, B. (2013) ‘Characterizing Architecturally Significant Requirements’, *IEEE Software*, 30(2), pp. 38–45. ISBN: 0740-7459 ISSN: 07407459 doi: 10.1109/MS.2012.174.

Choromanska, A. *et al.* (2015) ‘The Loss Surfaces of Multilayer Networks’, *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 38. Available at: http://proceedings.mlr.press/v38/choromanska15.pdf.

Crnkovic-Friis, L. (2018) *The Essential AI Handbook for Leaders*. Edited by A. Modin and J. Andrén. Göterborg: Peltarion AB. ISBN: 978-91-639-7114-3 Available at: http://pages.peltarion.com/rs/842-GTD-459/images/AI-handbook-desktop-PDF.pdf.

le Cun, Y. (1989) ‘Generalization and Network Design Strategies’, in Pfeifer, R. et al. (eds) *Connectionism in Perspective*. Elsevier. Available at: http://yann.lecun.com/exdb/publis/pdf/lecun-89.pdf.

Daróczy, G. (2010) ‘Artificial Intelligence and Cognitive Psychology’, in *Proceedings of the 8th International Conference on Applied Informatics*. Eger, Hungary, pp. 61–69. Available at: http://icai.ektf.hu/pdf/ICAI2010-vol1-pp61-69.pdf.

Das, S. R. (2016) *Data Science: Theories, Models, Algorithms, and Analytics*. S. R. Das. Available at: http://algo.scu.edu/%18sanjivdas/.

Department of Defense (2001) 'Systems Engineering Fundamentals'. ISBN: 0780379527 ISSN: 1346-1354 doi: 10.1016/j.cmpb.2010.05.002.

Dietterich, T. G. (1997) 'Machine Learning Research: Four Current Directions', *AI Magazine*, 18(4), pp. 97–136. Available at: http://web.engr.oregonstate.edu/~tgd/publications/aimag-survey.pdf.

Electronic Industries Alliance (1999) 'EIA STANDARD Processes for Engineering a System, EIA-632', *Journal of Equine Veterinary Science*, p. 120. ISSN: 07370806 doi: 10.1016/S0737-0806(99)80290-1.

European Commission (2018) *Nuclear Energy | Energy*. Available at: https://ec.europa.eu/energy/en/topics/nuclear-energy (Accessed: 18 February 2019).

Eysenck, M. W. and Keane, M. T. (2010) *Cognitive Psychology*. 6th edn. New York: Psychology Press. ISBN: 978-1-84169-539-6 .

Fischler, M. A. and Firschein, O. (1987) 'Intelligence - The Eye, the Brain, and the Computer'. Addison-Wesley Publishing Company, Inc. ISBN: 0-201-12001-1 .

Forsberg, K. and Mooz, H. (1991) 'The Relationship of System Engineering to the Project Cycle'. ISBN: 2334-5837 ISSN: 23345837 doi: 10.1002/j.2334-5837.1991.tb01484.x.

Forsberg, K., Mooz, H. and Cotterman, H. (2005) *Visualizing Project Management*. 3rd edn, *Project Management Journal*. 3rd edn. John Wiley & Sons, Inc. ISBN: 0471648485 .

Fortum (2018) *ADLAS - Advanced Licensing and Safety Design method for Nuclear Facilities | Fortum*. Available at: https://www.fortum.com/products-and-services/power-plant-services/nuclear-services/nuclear-new-builds/adlas-advanced (Accessed: 4 September 2018).

Geirhos, R. *et al.* (2019) 'ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness', *Proceedings of the 7th International Conference on Learning Representation (ICLR) 2019*. Available at: http://arxiv.org/abs/1811.12231.

van Gerven, M. and Bohte, S. (2018) *Artificial Neural Networks as Models of Neural Information Processing*. Lausanne: Frontiers Media. ISBN: 978-2-88945-401-3 doi: 10.3389/978-2-88945-401-3.

Ghahramani, Z. (2004) 'Unsupervised Learning', in *Advanced Lectures on Machine Learning*. 3176th edn. Berlin, Heidelberg: Springer, pp. 72–112. ISBN: 978-3-540-28650-9 doi: https://doi.org/10.1007/978-3-540-28650-9_5.

Glorot, X., Bordes, A. and Bengio, Y. (2011) 'Deep Sparse Rectifier Neural Network', *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 15, pp. 315–323. Available at: http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf.

Goddard, P. L. (1996) ‘A Combined Analysis Approach To Assessing Requirements For Safety Critical Real-Time Control Systems’, *Proceedings of the Annual Reliability and Maintainability Symposium*, pp. 110–115. ISBN: 0-7803-3112-5 ISSN: 0149144X .

Goldberg, Y. (2015) ‘A Primer on Neural Network Models for Natural Language Processing’, *Journal of Artificial Intelligence Research*, 57, pp. 345–420. ISBN: 9781627052986 ISSN: 10769757 doi: 10.1162/COLI_r_00312.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. MIT Press. ISBN: 9780262035613 Available at: http://www.deeplearningbook.org/.

Gori, M. and Tesi, A. (1992) ‘On the Problem of Local Minima in Backpropagation’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1), pp. 76–86. doi: 10.1109/34.107014.

Gray, A. *et al.* (2017) ‘Guide to Life Cycles and Life Cycle Models’. INCOSE UK Ltd, INCOSE and the Association for Project Management. Available at: https://www.apm.org.uk/media/13835/guide-to-lifecycle-models.pdf.

HAEA (2015) ‘Annex 3/A to Government Decree No. 118/2011: Design Requirements for new Nuclear Power Plant Units.’ Hungarian Atomic Energy Authority, pp. 1–105.

Han, X. *et al.* (2004) ‘Accuracy Improvement of Automatic Text Classification Based on Feature Transformation and Multi-Classifier Combination’, in Chi, C.-H. and Lam, K.-Y. (eds) *Content Computing*. Germany: Springer-Verlag Berlin Heidelberg, pp. 463–468. ISBN: 3-540-23898-0 Available at: https://link.springer.com/chapter/10.1007/978-3-540-30483-8_57.

Hancock, B. *et al.* (2018) ‘Training Classifiers with Natural Language Explanations’. Available at: http://arxiv.org/abs/1805.03818.

Haskins, B. *et al.* (2004) ‘Error Cost Escalation Through the Project Life Cycle’, *INCOSE-Annual Conference Symposium Proceedings- 14th Annual International Symposium*, pp. 1–8. ISBN: 2010003667 ISSN: 23345837 doi: 10.1002/j.2334-5837.2004.tb00608.x.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning*. New York: Springer. ISBN: 978-0-387-84858-7 doi: 10.1007/978-0-387-84858-7.

Hatcliff, J. *et al.* (2014) ‘Certifiably Safe Software-Dependent Systems: Challenges and Directions’, *Proceedings of the on Future of Software Engineering - FOSE 2014*, (May), pp. 182–200. ISBN: 978-1-4503-2865-4 Available at: http://dl.acm.org/citation.cfm?doid=2593882.2593895.

Hinton, G., Rumelhart, D. and Williams, R. (1986) ‘Learning Internal Representations by Error Propagation’, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge, Massachusetts: MIT Press.

Hochreiter, S. and Schmidhuber, J. (1997) ‘Long Short-Term Memory’, *Neural Computation*, 9(8), pp. 1735–1780. ISBN: 08997667 (ISSN) ISSN: 0899-7667 doi: 10.1162/neco.1997.9.8.1735.

Hull, E., Jackson, K. and Dick, J. (2011) *Requirements Engineering*. 3rd edn. Springer. ISBN: 978-1-84996-404-3 doi: 10.1007/978-1-84996-405-0.

Huyck, C. R. and Abbas, F. (2000) 'Natural Language Processing and Requirements Engineering: a Linguistics Perspective', in *Proceedings of 1st Asia-Pacific Conference on Software Quality*. Hong Kong, China, pp. 1–11. Available at: https://www.researchgate.net/publication/228749077_Natural_Language_Processing_and_Requirements_Engineering_a_Linguistics_Perspective.

IAEA (1989) *The Statute of the IAEA*. Available at: https://www.iaea.org/about/statute (Accessed: 3 September 2018).

IAEA (2000) *Operational Limits and Conditions and Operating Procedures for Nuclear Power Plants*. NS-G.2.2, *IAEA Safety Standards Series*. NS-G.2.2. ISBN: 92–0–102000–7 Available at: https://www-pub.iaea.org/MTCD/Publications/PDF/Pub1100_scr.pdf.

IAEA (2006a) *IAEA Safety Standards: Fundamental Safety Principles*. International Atomic Energy Agency. ISBN: 92–0–110706–4 Available at: https://www-pub.iaea.org/MTCD/publications/PDF/Pub1273_web.pdf.

IAEA (2006b) 'Strengthening the Global Nuclear Safety Regime'. Vienna: International Atomic Energy Agency. ISBN: 92–0–111306–4 Available at: https://www-pub.iaea.org/MTCD/Publications/PDF/Pub1277_web.pdf.

IAEA (2010) 'Licensing Process for Nuclear Installations'. Vienna: International Atomic Energy Agency. ISBN: 978–92–0–107510–9 doi: 10.13140/RG.2.1.3042.9289.

IAEA (2012) 'IAEA Nuclear Energy Series Project Management in Nuclear Power Plant Construction : Guidelines and Experience'. Vienna: International Atomic Energy Agency. ISBN: 978–92–0–122210–7 Available at: https://www.iaea.org/publications/8759/project-management-in-nuclear-power-plant-construction-guidelines-and-experience.

IAEA (2016) *Nuclear Power Reactors in the World*, *Agency, International Atomic Energy*. Vienna. ISBN: 978–92–0–103716–9 ISSN: 1011–2642 Available at: https://www-pub.iaea.org/MTCD/Publications/PDF/RDS_2-36_web.pdf.

IAEA (2019a) *Nuclear Security Series | IAEA*. Available at: https://www.iaea.org/resources/nuclear-security-series (Accessed: 18 February 2019).

IAEA (2019b) *Safety standards | IAEA*. Available at: https://www.iaea.org/resources/safety-standards (Accessed: 18 February 2019).

IEA and NEA (2015) *Technology Roadmap Annex: Nuclear Energy Case Studies*. Available at: https://www.oecd-nea.org/pub/techroadmap/techroadmap-2015-annex.pdf.

Ikonomakis, E., Kotsiantis, S. and Tampakas, V. (2005) 'Text Classification Using Machine Learning Techniques', *WSEAS TRANSACTIONS on COMPUTERS*, 4(8), pp. 966–974. Available at: https://www.researchgate.net/publication/228084521_Text_Classification_Using_Machine_Learning_Techniques.

INCOSE (2015) *Systems Engineering Handbook*. 4th edn. Edited by T. M. S. David D. Walden, Garry J. Roedler, Kevin J. Forsberg, R. Douglas Hamelin. Hoboken, New Jersey: John Wiley & Sons, Inc. ISBN: 9781118999400 .

ISO/IEC/IEEE (2010) 'ISO/IEC/IEEE 24765:2010 Systems and software engineering - Vocabulary', p. 410. ISBN: 9780738162058 doi: 10.1109/IEEESTD.2010.5733835.

ISO/IEC/IEEE (2011a) 'ISO/IEC/IEEE 29148 Systems and software engineering - Life cycle processes - Requirements engineering', p. 95. ISBN: 978-0-7381-6591-2 .

ISO/IEC/IEEE (2011b) 'ISO/IEC/IEEE 42010 Systems and Software Engineering - Architecture Description'. Geneva, p. 37. ISBN: 978-0-7381-7142-5 STD97174 Available at: https://www.iso.org/standard/50508.html.

ISO/IEC/IEEE (2015) 'ISO/IEC/IEEE 15288 Systems and software engineering - System life cycle processes'. ISBN: 978-0-7381-9532-2 .

ISO/IEC/IEEE (2016) 'ISO/IEC/IEEE 24748 Systems and software engineering - Life cycle management'. ISBN: 978-1-5044-0817-2 .

Jarrett, K. *et al.* (2009) 'What is the Best Multi-Stage Architecture for Object Recognition?', *Proceedings of the 12th International Conference on Computer Vision (ICCV)*, pp. 2146–2153. ISBN: 978-1-4244-4419-9 Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5459469.

Jung, A. (2018) 'Machine Learning : Basic Principles'. Available at: https://arxiv.org/pdf/1805.05052.pdf.

Jung, Y. *et al.* (2015) 'Integrated cost and schedule control systems for nuclear power plant construction: Leveraging strategic advantages to owners and EPC firms', *Science and Technology of Nuclear Installations*, 2015. ISSN: 16876083 doi: 10.1155/2015/190925.

Kafrawy, P. El, Mausad, A. and Esmail, H. (2015) 'Experimental Comparison of Methods for Multi-Label Classification in Different Application Domains', *International Journal of Computer Applications*, 114(19), pp. 1–9. ISSN: 09758887 doi: 10.5120/20083-1666.

Karstila, K. (2013) 'The Two Views To YVL-Information: Document View And Object View'. Teollisuuden Voima Oyj. doi: DOHA-#1513694.

Kessler, B., Numberg, G. and Hinrich, S. (1997) 'Automatic Detection of Text Genre', *Proceeding ACL '98/EACL '98 Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pp. 32–38. doi: 10.3115/976909.979622.

Keyes, J. (2015) *The CIO's Guide to Oracle Products and Solutions*. ISBN: 9781482249958 Available at: http://books.google.es/books?hl=es&lr=&id=YZpBBAAAQBAJ&oi=fnd&pg=PP1&dq=%22marketing%22+%2B+%22competitive+intelligence%22+%2B+%22apps%22&ots=Yx8LIl8Rgh&sig=Jm4K70Bu96bP-FsexecZFxJePTA.

Khan, S. U. R. *et al.* (2018) 'Temporal specificity-based text classification for information retrieval', *Turkish Journal of Electrical Engineering & Computer Sciences*, 26(6), pp. 2915–2926. ISSN: 13000632 doi: 10.3906/elk-1711-136.

Kocaleva, M. *et al.* (2016) 'Pattern Recognition and Natural Language Processing: State of the Art', *TEM Journal*, 5(2). doi: 10.18421/TEM52-18.

Korhonen, J. and Nuutinen, P. (2016) 'Safety Critical System'. Finland: World Intellectual Property Organization. Available at: https://patentimages.storage.googleapis.com/af/2b/6e/9a5cfa472cf8d9/WO2016120532A1.pdf.

Kriesel, D. (2007) 'A Brief Introduction to Neural Networks'. ISBN: 1476-4679 (Electronic)n1465-7392 (Linking) ISSN: 1432-0711 Available at: http://www.dkriesel.com.

Kumar, R. and Indrayan, A. (2011) 'Receiver Operating Characteristic (ROC) Curve for Medical Researchers', in *Indian Pediatrics*. Springer-Verlag, pp. 277–287. ISSN: 0974-7559 doi: https://doi.org/10.1007/s13312-011-0055-4.

Lamba, S. *et al.* (2014) 'Impact of Teaching Time on Attention and Concentration', *IOSR Journal of Nursing and Health Science (IOSR-JNHS)*, 3(4), pp. 01–04. Available at: http://www.iosrjournals.org/iosr-jnhs/papers/vol3-issue4/Version-1/A03410104.pdf.

Leffingwell, D. and Widrig, D. (1999) *Managing Software Requirements - A Unified Approach*, *Addison-Wesley*. Addison-Wesley. ISBN: 9780201615937 .

Legg, S. and Hutter, M. (2006) 'A Collection of Definitions of Intelligence', pp. 1–11. ISBN: 9781586037581 ISSN: 0922-6389 doi: 10.1207/s15327051hci0301_2.

Leopold, E. and Kindermann, J. (2002) 'Text Categorization with Support Vector Machines: How to Represent Texts in Input Space?', *Machine Learning*, 46, pp. 423–444. Available at: https://doi.org/10.1023/A:1012491419635.

Loukides, M. (2010) *What is data science?*, *O'Really Media*. Available at: https://www.oreilly.com/ideas/what-is-data-science (Accessed: 26 November 2018).

Lutz, S. T. and Huitt, W. G. (2003) 'Information Processing and Memory: Theory and Applications', *Educational Psychology Interactive*, pp. 1–17. Available at: http://www.edpsycinteractive.org/topics/cognition/infoproc.html.

Maggini, M., Rigutini, L. and Turchi, M. (2004) 'Pseudo-Supervised Clustering for Text Documents', *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence, WI 2004*, pp. 363–369. doi: 10.1109/WI.2004.10138.

Mandal, S., Kandar, S. and Ray, P. (2011) 'Open Incremental Model A Open Source Software Development Life Cycle Model "OSDLC"', *International Journal of Computer Applications*, 21(1), pp. 33–39. doi: 10.5120/2473-3327.

Marasco, J. (2007) *What Is the Cost of a Requirement Error?*, *StickyMinds*. Available at: https://www.stickyminds.com/article/what-cost-requirement-error (Accessed: 1 November 2018).

Di Martino, B. and Cretella, G. (2013) *Trustworthy Eternal Systems via Evolving Software, Data and Knowledge*, *Communications in Computer and Information Science*. Edited by A. Moschitti and B. Plank. Montpellier: Springer-Verlag Berlin Heidelberg. ISBN: 978-3-642-45260-4 ISSN: 1865-0937 doi: 10.1007/978-3-642-45260-4.

Martins, L. E. G. and Gorschek, T. (2016) 'Requirements Engineering for Safety-Critical Systems: A Systematic Literature Review', *Information and Software Technology*. Elsevier B.V., 75, pp. 71–89. ISSN: 0950-5849 doi: 10.1016/j.infsof.2016.04.002.

Mavin, A. *et al.* (2009) 'EARS (Easy Approach to Requirements Syntax)', *Proceedings of the IEEE International Conference on Requirements Engineering*, (October), pp. 317–322. ISSN: 1090705X doi: 10.1109/RE.2009.9.

McCarthy, J. *et al.* (1955) 'A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence'. ISBN: 9780874216561 ISSN: 00225355 Available at: http://www-formal.stanford.edu/jmc/history/dartmouth.pdf.

McCarthy, J. (2007) 'What Is Artificial Intelligence'. Available at: http://www-formal.stanford.edu/jmc/whatisai.html.

McCulloch, W. S. and Pitts, W. (1943) 'A Logical Calculus of the Ideas Immanent in Nervous Activity', *Bulletin of Mathematical Biophysics*, 5(4), pp. 115–133. ISSN: 1522-9602 Available at: https://doi.org/10.1007/BF02478259.

Merity, S. (2017) *The WikiText Long Term Dependency Language Modeling Dataset*. Available at: https://blog.einstein.ai/the-wikitext-long-term-dependency-language-modeling-dataset/ (Accessed: 15 February 2019).

Miailhe, N. and Hodes, C. (2017) 'The Third Age of Artificial Intelligence', *Field Actions Science Reports*, (Special Issue 17), pp. 6–11. ISSN: 1867-8521 Available at: http://journals.openedition.org/factsreports/4383.

Ministry of Economic Affairs and Employment (1987) 'Nuclear Energy Act (990/1987)'. The Ministry of Economic Affairs and Employment. Available at: https://www.finlex.fi/fi/laki/kaannokset/1987/en19870990_20080342.pdf.

MIT (2018) 'The Future of Nuclear Energy in a Carbon-Constrained World', *MIT Future of Series*. Massachusetts Institute of Technology, pp. 1–275. doi: 10.1021/es50002a013.

Mohammadi, M. *et al.* (2019) 'CS 224n: Natural Language Processing with Deep Learning'. Stanford, California: Stanford University, pp. 1–14. Available at: http://web.stanford.edu/class/cs224n/readings/cs224n-2019-notes05-LM_RNN.pdf.

Monner, D. and Reggia, J. A. (2012) 'A Generalized LSTM-like Training Algorithm for Second-Order Recurrent Neural Networks', *Neural Networks*, 25(301), pp. 70–83. doi: 10.1016/j.neunet.2011.07.003.

Mor-Yosef, S. *et al.* (1990) 'Ranking the Risk Factors for Cesarean: Logistic Regression Analysis of a Nationwide Study', *Obstetrics and Gynaecology*, pp. 944–947. ISSN: 0029-7844 Available at: https://www.researchgate.net/publication/20812830_Ranking_the_risk_factors_for_cesarean_Logistic_regression_analysis_of_a_nationwide_study.

Nakashima, H. (1999) 'AI as Complex Information Processing', in *Minds and Machines*. Kluwer Academic Publishers, pp. 57–80. ISBN: 1572-8641 doi: https://doi.org/10.1023/A:1008322730047.

NASA (2016) *Systems Engineering Handbook*. ISBN: SP-2016-6105 Available at: https://www.nasa.gov/sites/default/files/atoms/files/nasa_systems_engineering_handbook_0.pdf.

Nielsen, M. A. (2015) *Neural Networks and Deep Learning*, *Determination Press*. Available at: http://neuralnetworksanddeeplearning.com/index.html.

Nuutinen, P., Sipola, S. and Rantakaulio, A. (2016) 'Advanced Licensing and Safety Engineering Method -ADLAS®', (November), pp. 2–3. ISBN: 9781510851160 .

Nuutinen, P., Sipola, S. and Rantakaulio, A. (2017) 'Advanced Licensing and Safety Engineering Method -ADLAS®'. ISBN: 9781510851160 .

O'Reilly, R. C. *et al.* (2012) *Computational Cognitive Neuroscience*. 3rd edn. Wiki Book. Available at: http://ccnbook.colorado.edu.

Onowakpo, J. and Ebbah, G. (2002) 'Deploying Artificial Intelligence Techniques in Software Engineering', *American Journal of Undergraduate Research*, 1(1), pp. 19–24.

ONR (2014) 'Safety Assessment Principles for Nuclear Facilities'. The Office for Nuclear Regulation. Available at: http://www.onr.org.uk/saps/index.htm.

ONR (2016) 'The Purpose, Scope and Content of Nuclear Safety Cases'. The Office for Nuclear Regulation, 4, pp. 1–23. Available at: http://www.onr.org.uk/operational/tech_asst_guides/ns-tast-gd-051.pdf.

Ormandjieva, O., Hussain, I. and Kosseim, L. (2007) 'Toward a Text Classification System for the Quality Assessment of Software Requirements Written in Natural Language', in *Proceeding of the Fourth International Workshop on Software Quality Assurance, SOQUA 2007, in conjunction with the 6th ESEC/FSE joint meeting*. Dubrovnik, Croatia. doi: 10.1145/1295074.1295082.

Pan, S. J. and Yang, Q. (2010) 'A Survey on Transfer Learning', *IEEE Transactions on Knowledge and Data Engineering*. IEEE, 22(10), pp. 1345–1359. doi: 10.1109/TKDE.2009.191.

Pushpankar, K. P. and Muktabh, M. S. (2017) 'Train Once, Test Anywhere: Zero-Shot Learning for Text Classification', *Proceedings of the 6th International Conference on Learning Representation (ICLR) 2018*, (2017), pp. 1–6. Available at: http://arxiv.org/abs/1712.05972.

Rojas, R. (1996) *Neural Networks - A Systematic Introduction*. 1st edn. Berlin: Springer-Verlag Berlin Heidelberg. ISBN: 978-3-642-61068-4 doi: 10.1007/978-3-642-61068-4.

Rosenblatt, F. (1961) *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. New York: Cornell Aeronautical Laboratory, Inc. ISSN: VG-1 196-G-8 .

Rothman, J. (2002) 'What Does It Cost to Fix a Defect?', *StickyMinds*. Available at: http://www.stickyminds.com/article/what-does-it-cost-fix-defect.

Royce, W. (1970) 'Managing the Development of large Software Systems', *IEEE Wescon*, (August), pp. 1–9. ISBN: 0-89791-216-0 .

Rumelhart, D. E. and McClelland, J. L. (1986) *Parallel Distributed Processing*. Edited by J. A. Feldman, P. J. Hayes, and D. E. Rumelhart. Cambridge, Massachusetts: MIT Press. ISBN: 0-262-13218-4 .

Saaksvuori, A. and Immonen, A. (2008) *Product Lifecycle Management*, *Springer Science & Business Media*. ISBN: 3540781730 doi: 10.1007/978-3-540-78172-1.

Samuel, A. L. (1959) 'Some Studies in Machine Learning Using the Game of Checkers', *IBM Journal of Research and Development*, 3(3), pp. 210–229. ISBN: 0018-8646 ISSN: 0018-8646 doi: 10.1147/rd.33.0210.

Saunders, M., Lewis, P. and Thornhill, A. (2007) *Research Methods for Business Students*. 4th edn. Edinburgh: Pearson Education Limited. ISBN: 978-0-273-70148-4 .

Schank, R. (1980) 'Where's the A.I?', *AI Magazine*, 12(4), p. 38. Available at: https://aaai.org/ojs/index.php/aimagazine/article/view/917/835.

Schneider, M. and Froggatt, A. (2018) *The World Nuclear Industry Status*. Paris, London. Available at: https://www.worldnuclearreport.org/IMG/pdf/20180902wnisr2018-hr.pdf.

Sebastiani, F. (2002) 'Categorization Machine Learning in Automated Text Categorization', *ACM Computing Serveys*, 34(1), pp. 1–47. ISBN: 0360-0300 ISSN: 15693511 doi: 10.1145/505282.505283.

SFS (2010) 'SFS-EN 61508-4 Functional safety of electrical/electronic/programmable electronic safety-related systems. Part 4: Definitions and abbreviations'. Finnish Standards Association SFS ry. Available at: https://www.sfs.fi/.

SFS (2015) 'SFS-EN ISO 9000 Quality management systems . Fundamentals and vocabulary (ISO 9000:2015)'. Finnish Standards Association SFS ry. Available at: https://www.sfs.fi/.

Sharma, S. and Pandey, S. K. (2013) 'Integrating AI Techniques in Requirements Phase: A Literature Review', *IJCA Proceedings on 4th International IT Summit Confluence 2013 - The Next Generation Information Technology Summit*, Confluence(2), pp. 21–25.

Singh, D., Thakur, A. and Chaudhary, A. (2015) 'A Comparative Study between Waterfall and Incremental Software Development Life Cycle Model', *International Journal of Emerging Trends in Science and Technology*, 2(04), pp. 2202–2208. Available at: http://ijetst.in/article/v2-i4/9 ijetst.pdf.

Smith, A. and Thangarajoo, Y. (2015) 'Lean Thinking: An Overview', *Industrial Engineering and Management*, 04(02), pp. 1–6. ISSN: 21690316 doi: 10.4172/2169-0316.1000159.

Sommerville, I. and Sawyer, P. (Pete H. . (1997) 'Requirements Engineering: A Good Practice Guide', p. 404. ISBN: 0471974447 ISSN: 0002-9890 doi: 10.2307/2695615.

STUK (2013a) 'Guide YVL B.1 Safety Design of a Nuclear Power Plant'. Helsinki: Radiation and Nuclear Safety Authority, p. 46. ISBN: 9789523090460 Available at: https://www.stuklex.fi/en/ohje/YVLB-1.

STUK (2013b) 'YVL A.6 Conduct of Operations at a Nuclear Power Plant'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-056-9 Available at: https://www.stuklex.fi/en/ohje/YVLA-6.

STUK (2013c) 'YVL B.2 Classification of Systems, Structures and Components of a Nuclear Facility'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-071-2 Available at: https://www.stuklex.fi/en/ohje/YVLB-2.

STUK (2013d) 'YVL B.5 Reactor Coolant Circuit of a Nuclear Power Plant'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-083-5 Available at: https://www.stuklex.fi/en/ohje/YVLB-5.

STUK (2013e) 'YVL E.3 Pressure Vessels and Piping of a Nuclear Facility'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-041-5 Available at: https://www.stuklex.fi/en/ohje/YVLE-3.

STUK (2013f) 'YVL E.6 Buildings and Structures of a Nuclear Facility'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-134-4 Available at: https://www.stuklex.fi/en/ohje/YVLE-6.

STUK (2013g) 'YVL E.7 Electrical and I&C Equipment of a Nuclear Facility'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-137-5 Available at: https://www.stuklex.fi/en/ohje/YVLE-7.

STUK (2013h) 'YVL E.8 Valves of a Nuclear Facility'. Helsinki: Radiation and Nuclear Safety Authority. ISBN: 978-952-309-140-5 Available at: https://www.stuklex.fi/en/ohje/YVLE-8.

STUK (2018a) *Regulatory Guides on nuclear safety (YVL)*. Available at: https://www.stuk.fi/web/en/regulations/stuk-s-regulatory-guides/regulatory-guides-on-nuclear-safety-yvl- (Accessed: 18 September 2018).

STUK (2018b) *STUK Regulations | Regulation | Stuklex*. Available at: https://www.stuklex.fi/en/maarays (Accessed: 18 September 2018).

STUK (2018c) *YVL-ohjevaatimusten attribuutit ja niiden käyttö*. doi: DOHA-#1512473.

Sun, T. (2009) 'Spam Filtering based on Naive Bayes Classification'. Available at: http://www.cs.ubbcluj.ro/~gabis/DocDiplome/Bayesian/000539771r.pdf.

Suthaharan, S. (2016) *Machine Learning Models and Algorithms for Big Data Classification*. 36th edn, *16th AIAA Computational Fluid Dynamics Conference*. 36th edn. Edited by R. Sharda and S. Voß. Springer. ISBN: 978-1-4899-7641-3 doi: 10.1007/978-1-4899-7641-3.

Sutton, R. S. and Barto, A. G. (1998) *Reinforcement Learning: An Introduction*. 1st edn. Cambridge, Massachusetts: MIT Press. ISBN: 9780262193986 Available at: https://mitpress.mit.edu/books/reinforcement-learning-second-edition.

Tamai, T. and Anzai, T. (2018) 'Quality Requirements Analysis with Machine Learning', *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2018)*, pp. 241–248. ISBN: 978-989-758-300-1 Available at: http://www.scitepress.org/Papers/2018/66945/66945.pdf.

Tan, B. *et al.* (2017) 'Distant Domain Transfer Learning', *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pp. 2604–2610. Available at: https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14446.

Torch Contributors (2018) *SOURCE CODE FOR TORCH.NN.MODULES.RNN*. Available at: https://pytorch.org/docs/stable/_modules/torch/nn/modules/rnn.html#LSTM (Accessed: 1 March 2019).

Tripathy, A., Agrawal, A. and Rath, S. K. (2014) 'Requirement Analysis using Natural Language Processing', in *Proceeding of the Fifth International Conference on Advances in Computer Engineering (ACE 2014), At Kochi, India*, pp. 39–45. Available at: https://www.researchgate.net/publication/272792284_Requirement_Analysis_using_Natural_Language_Processing.

Tyugu, E. (2007) *Algorithms and Architectures of Artificial Intelligence*. IOS Press. ISBN: 9781614997726 ISSN: 09226389 Available at: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85026754650&partnerID=40&md5=f4f321592ae9d35779c4457309ba66ac.

Vujica Herzog, N. and Tonchia, S. (2014) 'An Instrument for Measuring the Degree of Lean Implementation in Manufacturing', *Journal of Mechanical Engineering.*, 60(12 OP-Strojniski Vestnik / Journal of Mechanical Engineering. Dec2014, Vol. 60 Issue 12, p797-803. 7p.), pp. 797–803. ISBN: 00392480 ISSN: 00392480 doi: 10.5545/sv-jme.2014.1873.

Wang, Y. *et al.* (2016) 'Attention-based LSTM for Aspect-level Sentiment Classification', *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 606–615. doi: 10.18653/v1/D16-1058.

Weizhong, Y. and Goebel, K. F. (2004) 'Designing Classifier Ensembles with Constrained Performance Requirements', *Proceedings of SPIE, Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, 5434, pp. 59–68. doi: 10.1117/12.542616.

Winkler, J. and Vogelsang, A. (2016) 'Automatic Classification of Requirements Based on Convolutional Neural Networks', *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference Workshops, REW 2016*. IEEE, pp. 39–45. doi: 10.1109/REW.2016.021.

Winston, P. H. (1993) *Artificial Intelligence*. Massachusetts: Addison-Wesley. ISBN: 0-201-53377-4 Available at: https://courses.csail.mit.edu/6.034f/ai3/rest.pdf.

World Nuclear Association (2013) *Licensing and Project Development of New Nuclear Plants*. London. Available at: http://www.world-nuclear.org/uploadedFiles/org/WNA/Publications/Working_Group_Reports/WNA_REPORT_Nuclear_Licensing.pdf.

World Nuclear Association (2016) *World Nuclear Performance Report 2016 - Asia Edition*. ISSN: 00385077 Available at: https://www.world-nuclear.org/press/press-statements/launch-of-world-nuclear-performance-report-2016-as.aspx.

World Nuclear Association (2018) 'World Nuclear Performance Report'. London: World Nuclear Association. ISBN: 978-1-4244-2949-3 ISSN: 2155-897 doi: 10.1109/PVSC.2009.5411432.

Xia, W. *et al.* (2018) 'Novel Architecture for Long Short-Term Memory Used in Question Classification', *Neurocomputing*, 299, pp. 20–31. ISSN: 18728286 doi: 10.1016/j.neucom.2018.03.020.

Yalla, P. and Sharma, N. (2015) 'Integrating Natural Language Processing and Software Engineering', *International Journal of Software Engineering and its Applications*, 9(11), pp. 127–136. doi: 10.14257/ijseia.2015.9.11.12.

Yuste, R. (2015) 'From the Neuron Doctrine to Neural Networks', *Nature Reviews Neuroscience*. Nature Publishing Group, 16, pp. 487–497. doi: http://dx.doi.org/10.1038/nrn3962.

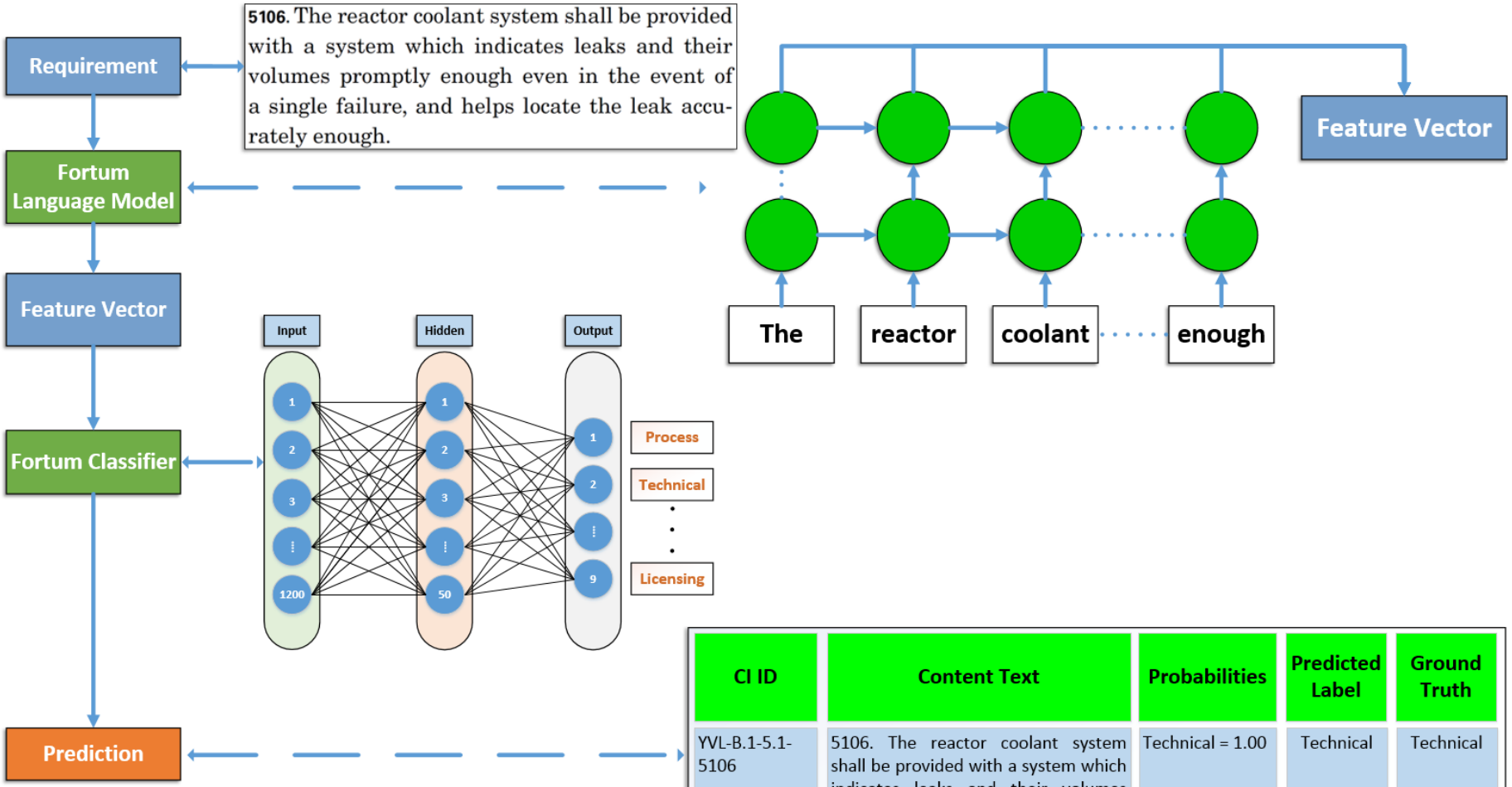
Zhou, J. T. *et al.* (2016) 'Transfer Learning for Cross-Language Text Categorization through Active Correspondences Construction', *Proceedings of 13th AAAI Conference on Artificial Intelligence*, pp. 2400–2406. Available at: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/download/11882/11890.

Zweig, M. H. and Campbell, G. (1993) 'Receiver-Operating Characteristic (ROC) Plots: A Fundamental Evaluation Tool in Clinical Medicine', *Clinical Chemistry*, 39(4), pp. 561–577. Available at: http://clinchem.aaccjnls.org/content/39/4/561.

Appendices

Appendix 1 Simplified and illustrative flowsheet of the testing procedure

Appendix 1 Simplified and illustrative flowsheet of the testing procedure



CI ID	Content Text	Probabilities	Predicted Label	Ground Truth
YVL-B.1-5.1-5106	5106. The reactor coolant system shall be provided with a system which indicates leaks and their volumes promptly enough even in the event of a single failure, and helps locate the leak accurately enough.	Technical = 1.00	Technical	Technical